Complex Critical Systems in Event-B

Yamine AIT-AMEUR
Neeraj Kumar SINGH

INPT-ENSEEIHT/IRIT
{yamine, nsingh}@toulouse-inp.fr

7 October 2021

Outline

Definition

A life-critical system or safety-critical system is a system whose failure or malfunction may result in serious injuries, loss of life, economical damage and environmental harm.

Critical System Failure

Systems Failure

- Therac-25 (1985-1987): six people overexposed through radiation.
- Pacemaker and ICD (1990-2002): 17,323 pacemakers and ICDs were explanted that includes 61 deaths.
- Insulin Infusion Pump (IIP) (2010): 5000 adverse events that includes 30 deaths.
- Missing Malaysian Plane MH370 (8 March, 2014): Unknown.
- Satellite Failure:+150: `http://www.sat-nd.com/failures/`.

The Cardiac Pacemaker (Grand Challenge)

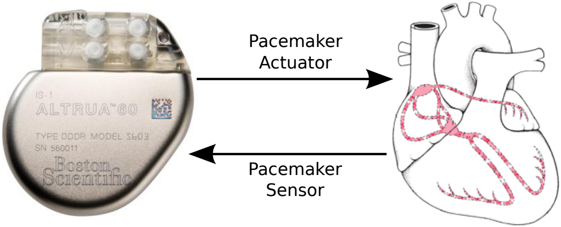
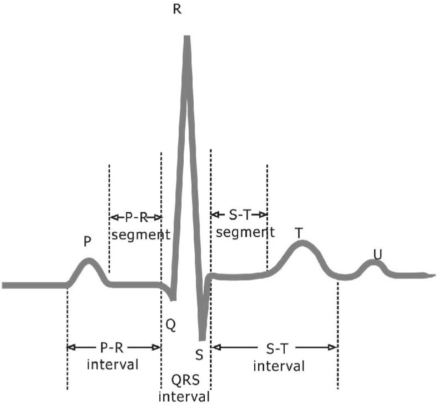


Figure: Closed-loop Modelling of the Heart & Cardiac Pacemaker

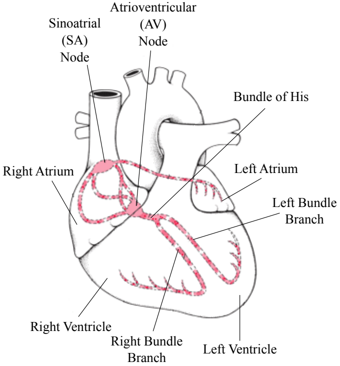
The Human Heart

The Heart Modeling

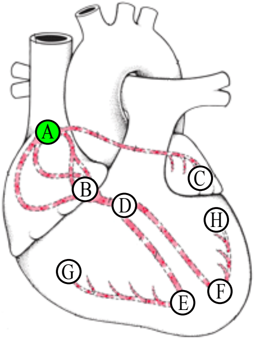
Electrocardiogram (ECG)



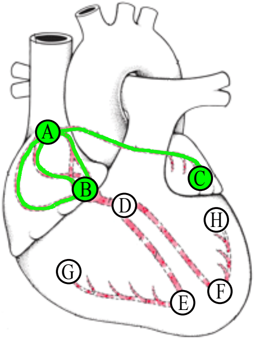
The Electrical Conduction



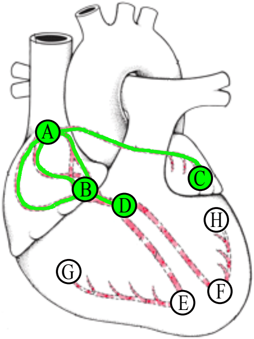
Heart Conduction Through Landmark Nodes



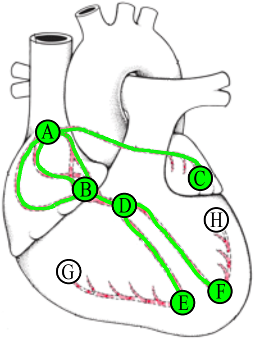
Heart Conduction Through Landmark Nodes



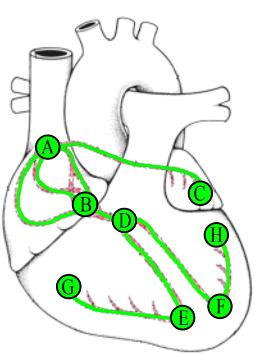
Heart Conduction Through Landmark Nodes



Heart Conduction Through Landmark Nodes



Heart Conduction Through Landmark Nodes



Heart Definition

Definition 1 (The Heart System)

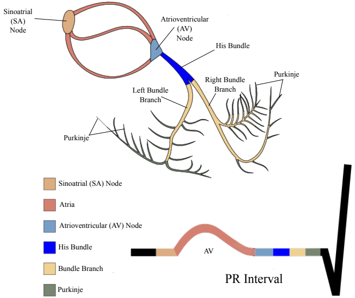
Given a set of nodes N, a transition (conduction) T is a pair (i, j), with $i, j \in N$ *. A transition is denoted by* $i \rightsquigarrow j$*.The heart system is a tuple* $HSys = (N,\ T,\ N_0,\ TW_{time},\ CW_{speed}\ )$ *where:*

- $N = \{\ A,\ B,\ C,\ D,\ E,\ F,\ G,\ H\ \}$ *is a finite set of landmark nodes;*
- $T \subseteq N \times N = \{A \mapsto B,\ A \mapsto C,\ B \mapsto D,\ D \mapsto E,\ D \mapsto F,\ E \mapsto G,\ F \mapsto H\}$ *is a set of transitions;*
- $N_0\ = A$ *is the initial landmark node;*
- $TW_{time} \in N \rightarrow TIME$ *is a weight function as time delay;*
- $CW_{speed} \in T \rightarrow SPEED$ *is a weight function as impulse propagation speed.*

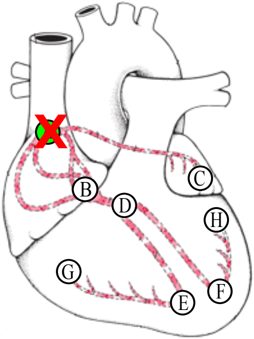
Time Intervals and Impulse Propagation in the ECG

Location in the heart	Impulse Propagation Time (ms.) **Property 1** (TW_{time})	Location in the heart	Impulse Propagation Speed (cm/sec.) **Property 2** (CW_{speed})
SA Node (A)	0..10	$A \mapsto B$	30..50
Left atria muscle fibers (C)	70..90	$A \mapsto C$	30..50
AV Node (B)	50..70	$B \mapsto D$	100..200
Bundle of His (D)	125..160	$D \mapsto E$	100..200
Right Bundle Branch (E)	145..180	$D \mapsto F$	100..200
Left Bundle Branch (F)	145..180	$E \mapsto G$	300..400
Right Purkinje fibers (G)	150..210	$F \mapsto H$	300..400
Left Purkinje fibers (H)	150..230		

Table: Cardiac Activation Time and Cardiac Velocity

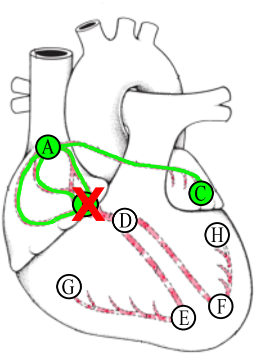


Heart Blocks (Abnormal Heart Rhythm)



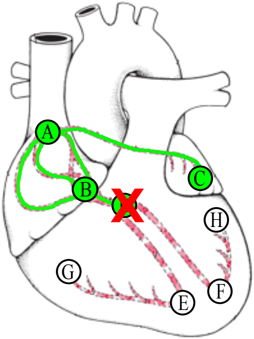
SA Block

Heart Blocks (Abnormal Heart Rhythm)



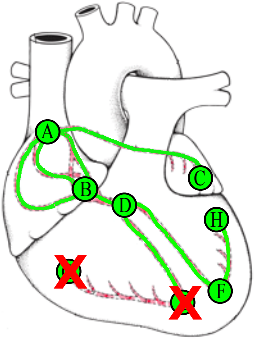
AV Block

Heart Blocks (Abnormal Heart Rhythm)



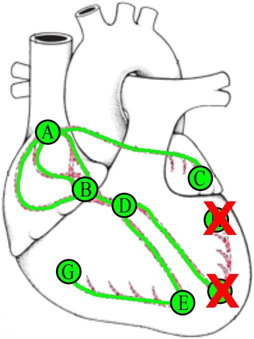
Infra-Hisian Block

Heart Blocks (Abnormal Heart Rhythm)



Right Bundle Branch Block

Heart Blocks (Abnormal Heart Rhythm)



Left Bundle Branch Block

2D Cellular Automata and State Transition Model

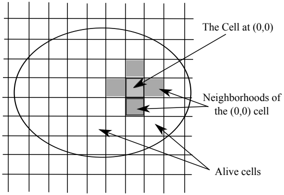
Definition 3 (State Transition of a Cell)

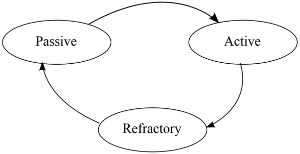
The heart muscle system is composed of heterogeneous cells, the cellular automata model of the muscle system, CAM_{CA}, is characterized with no dependencies on the type of cells. CAM_{CA} is defined as follows:

$CAM_{CA} = < S, N, T >$
$S = \{Active, Passive, Refractory\}$
$N_{m,n} = \{(m,n),(m+1,n),(m-1,n),(m,n+1),(m,n-1)\}$
$T : S^{|N|} \rightarrow S$

$$s_{t+1}(m,n) = \begin{cases} Refractory & \text{if } s_t(m,n) = Active \\ Passive & \text{if } s_t(m,n) = Refractory \\ Active & \text{if } s_t(m,n) = Passive \text{ and any neighbor is in } Active \text{ state} \\ Passive & \text{if } s_t(m,n) = Passive \text{ and none neighbor is in } Active \text{ state} \end{cases}$$

where, $s_t(m,n)$ denotes the state of the cell located at (m,n).





Development in Event-B

Abstract Model and Chain of Refinements

- Abstract Model
- Refinement 1: Introducing Steps in the Propagation
- Refinement 2: Impulse Propagation
- Refinement 3: Perturbation the Conduction
- Refinement 4: Getting a Cellular Model

Proof Statistics

Model	Total number of POs	Automatic Proof	Interactive Proof
Abstract Model	29	22(76%)	7(24%)
First Refinement	9	6(67%)	3(33%)
Second Refinement	159	155(97%)	4(3%)
Third Refinement	10	1(10%)	9(90%)
Fourth Refinement	11	10(91%)	1(9%)
Total	218	194(89%)	24(11%)

The Cardiac Pacemaker

The Cardiac Pacemaker

Pacemaker

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

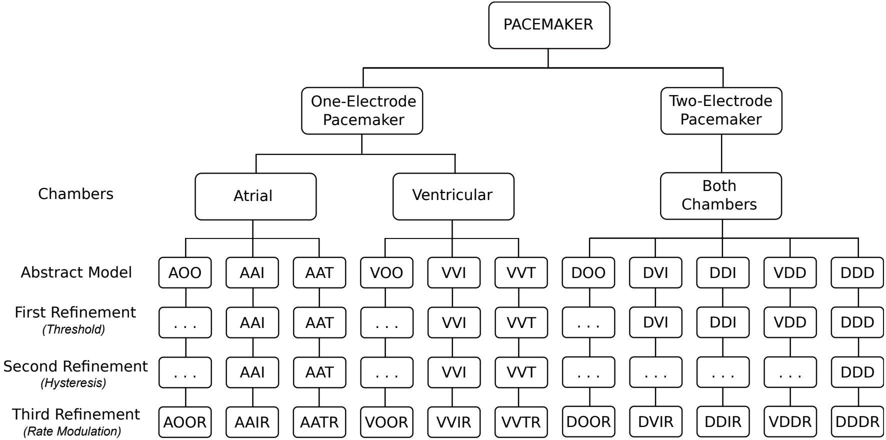
The Cardiac Pacemaker

Operating Modes : NASPE/BPEG Generic Code

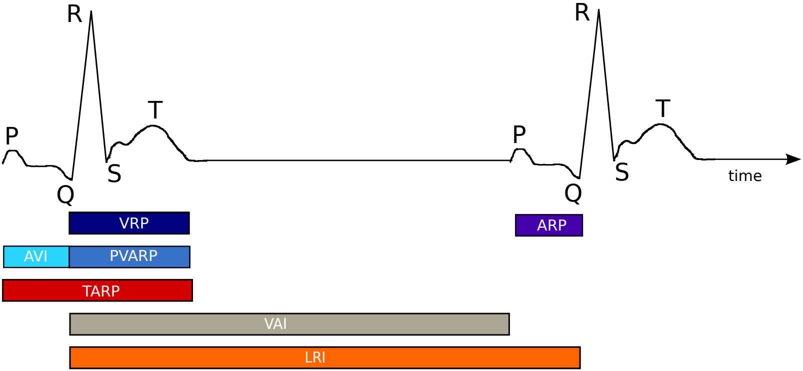
Category	Chambers Paced	Chambers Sensed	Response to Sensing	Rate Modulation
Letters	**O**-None **A**-Atrium **V**-Ventricle **D**-Dual(A+V)	**O**-None **A**-Atrium **V**-Ventricle **D**-Dual(A+V)	**O**-None **T**-Triggered **I**-Inhibited **D**-Dual(T+I)	**R**-Rate Modulation

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.)

Hierarchical Development of Operating Modes



Timing Cycles

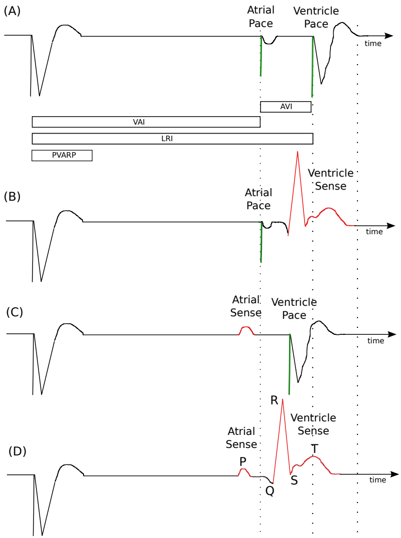


VRP - Ventricular Refractory Period
ARP - Atrial Refractory Period
AVI - Atrioventricular Interval
PVARP - Post Ventricular Atrial Refractory Period
TARP - Total Atrial Refractory Period
VAI - Ventriculoatrial Interval
LRI - Lower Rate Interval
URI - Upper Rate Interval

General Requirements (Boston Scientific)

1. The bradycardia operating modes (AOO, AAT, VVI, ... , DDDR) shall be programmable.
2. The Pacemaker shall support single and dual chamber pacing modes.
3. The device shall actuate for pacing in the heart with programmable voltages and widths.
4. The pacing pulse amplitudes and pacing width shall be programmable for each chamber (e.g. atrial, ventricular).
5. The device shall support rate adaptive pacing in order to meet the physiological needs.

The DDD Pacing Scenarios



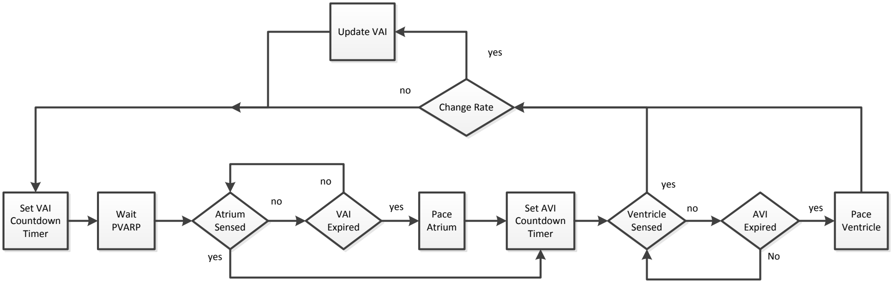
DOO and DDD Modes Requirements

1. The pacemaker in DOO mode must pace in both atria and ventricular chambers without sensing any intrinsic activities from both chambers according to the programmable parameters.
2. In DDD mode, when the pacing system senses intrinsic activities in both chambers, the pacemaker must be triggered or inhibited to pace in both chambers according to the programmable parameters.

Rate Adaptive Modes (AOOR, DOOR, ..., DDDR) Requirements

1. The pacemaker in rate adaptive mode must pace and sense in any chamber according to the selected mode by changing the rate to meet the physical needs according to the programmable parameters.

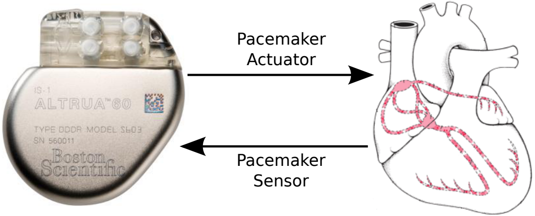
Flowchart for DDDR Pacing Cycle



Formal Development

Closed-loop Model (Heart & Cardiac Pacemaker)

- Simple Pacemaker Model
- Closed-loop Model



- Abstract Model: Pacing and Sensing Activities + Normal and Abnormal Heart Behaviour.
- Refinement 1: *Threshold* + Impulse Propagation .
- Refinement 2: Hysteresis + Perturbation the Conduction.
- Refinement 3: Rate Modulation + Cellular Model.

Formalization of the Closed-loop model: Abstract Model

$axm1 : partition(ConductionNode, \{A\}, \{B\}, \{C\}, \{D\}, \{E\}, \{F\}, \{G\}, \{H\})$
$axm2 : ConductionTime \in ConductionNode \rightarrow \mathbb{P}(0 \,..\, 230)$
$axm3 : ConductionPath \subseteq ConductionNode \times ConductionNode$
$axm4 : ConductionSpeed \in ConductionPath \rightarrow \mathbb{P}(5 \,..\, 400)$
$axm5 : LRL \in 30 \,..\, 175 \wedge URL \in 50 \,..\, 175 \wedge PVARP \in 150 \,..\, 500$
$axm6 : ARP \in 150 \,..\, 500 \wedge VRP \in 150 \,..\, 500 \wedge status = \{ON, OFF\}$

$inv1 : ConductionNodeState \in ConductionNode \rightarrow BOOL$
$inv2 : CConductionTime \in ConductionNode \rightarrow 0 \,..\, 300$
$inv3 : CConductionSpeed \in ConductionPath \rightarrow 0 \,..\, 500$
$inv4 : HeartState \in BOOL$
$inv5 : PM\_Actuator\_A \in status \wedge PM\_Actuator\_V \in status$
$inv6 : PM\_Sensor\_A \in status \wedge PM\_Sensor\_V \in status$
$inv7 : Pace\_Int \in URI \,..\, LRI \wedge sp \in 1 \,..\, Pace\_Int$
$inv8 : sp < VRP \wedge sp < PVARP$
$\Rightarrow$
$PM\_Actuator\_V = OFF \wedge PM\_Sensor\_A = OFF \wedge$
$PM\_Sensor\_V = OFF \wedge PM\_Actuator\_A = OFF$

$inv9 : PM\_Actuator\_V = ON \Rightarrow sp = Pace\_Int$
$inv10 : PM\_Actuator\_A = ON \Rightarrow (sp \geq Pace\_Int - FixedAV)$
...

DDD Pacing Modes

EVENT Actuator_ON_V
WHEN
grd1 : $PM\_Actuator\_V = OFF$
grd2 : $sp = Pace\_Int$
grd3 : $sp \geq VRP \wedge sp \geq PVARP$
THEN
act1 : $PM\_Actuator\_V := ON$
act2 : $last\_sp := sp$
END

EVENT Actuator_OFF_V
WHEN
grd1 : $PM\_Actuator\_V = ON$
grd2 : $sp = Pace\_Int$
grd3 : $PM\_Actuator\_A = OFF$
grd4 : $PM\_Sensor\_A = OFF$
THEN
act1 : $PM\_Actuator\_V := OFF$
act2 : $AV\_Count := 0$
act3 : $PM\_Sensor\_V := OFF$
act4 : $sp := 1$
END

EVENT tic
WHEN
grd1 : $sp < Pace\_Int$
THEN
act1 : $sp := sp + 1$
END

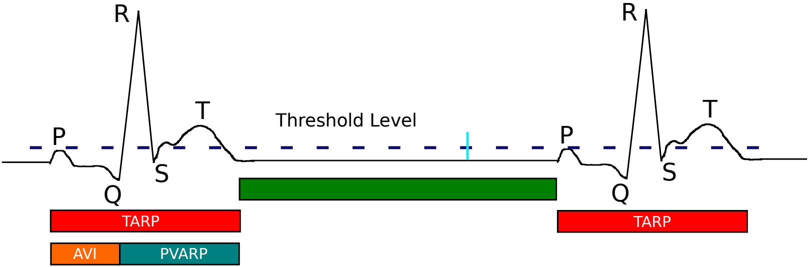
Continue...

EVENT HeartOK
WHEN
grd1 : $\forall i \cdot i \in ConductionNode \Rightarrow ConductionNodeState(i) = TRUE$
grd2 : $\forall i \cdot i \in ConductionNode \Rightarrow CConductionTime(i) \in ConductionTime(i)$
grd3 : $\forall i, j \cdot i \mapsto j \in ConductionPath \Rightarrow$
$CConductionSpeed(i \mapsto j) \in ConductionSpeed(i \mapsto j)$
THEN
act1 : $HeartState := TRUE$
END

EVENT Sensor_ON_V
WHEN
grd1 : $PM\_Sensor\_V = OFF$
grd2 : $(sp \geq PVARP \wedge sp > VRP$
grd3 : $sp > Pace\_Int - FixedAV$
grd4 : $PM\_Actuator\_A = OFF$
THEN
act1 : $PM\_Sensor\_V := ON$
END

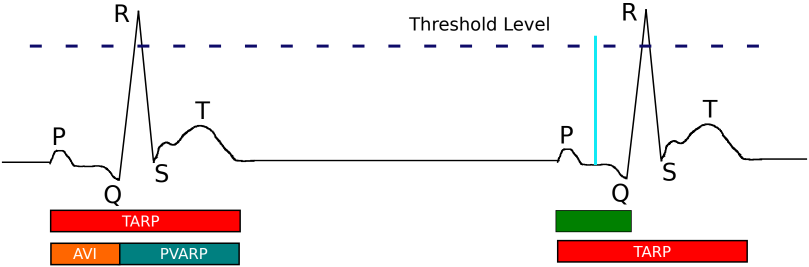
EVENT Sensor_OFF_V
WHEN
grd1 : $PM\_Sensor\_V = ON$
grd2 : $sp \geq VRP \wedge sp \geq PVARP$
grd3 : $sp \geq Pace\_Int - FixedAV$
grd4 : $PM\_Actuator\_V = OFF$
THEN
act1 : $PM\_Sensor\_V := OFF$
END

First Refinement (Threshold): Sensor Activity in DDD



$$
\begin{aligned}
&inv1 : sp > PVARP \land sp < Pace\_Int - FixedAV \\
&\quad \Rightarrow \\
&\quad PM\_Sensor\_A = ON \land \\
&\quad PM\_Sensor\_V = OFF \land \\
&\quad PM\_Actuator\_A = OFF \land \\
&\quad PM\_Actuator\_V = OFF
\end{aligned}
$$

First Refinement (Threshold): Sensor Activity in DDD



$$
\begin{aligned}
inv1: \; & sp > Pace\_Int - FixedAV \land sp < Pace\_Int \\
& \Rightarrow \\
& PM\_Sensor\_V = ON \land \\
& PM\_Sensor\_A = OFF \land \\
& PM\_Actuator\_A = OFF \land \\
& PM\_Actuator\_V = OFF
\end{aligned}
$$

What is Hysteresis?

Hysteresis is a programmed feature whereby the pacemaker paces at a faster rate than the sensing rate and it provides consistent pacing to the atrial or ventricle, or prevents the constant pacing of the atrial or ventricle. The main purpose of hysteresis is to allow the patient to have his or her own underlying rhythm as much as possible.

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 : $Hyt\_Pace\_Int \in Pace\_Int \,..\, LRI$
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 : $acler\_sensed \geq acc\_thr \wedge acler\_sensed\_flag = TRUE \Rightarrow Pace\_Int = 60000/MSR$

Model Validation & Analysis

ProB Model Checker

The ProB model checker is used for,

- to verify the developed system requirements;
- to check the required behaviour of the pacemaker for each operating mode;
- deadlock checking;
- to discover the counter examples;

Simple Model Vs. Closed-loop Model

Model	Total number of POs	Automatic Proof	Interactive Proof
Simple 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%)
Simple 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%)

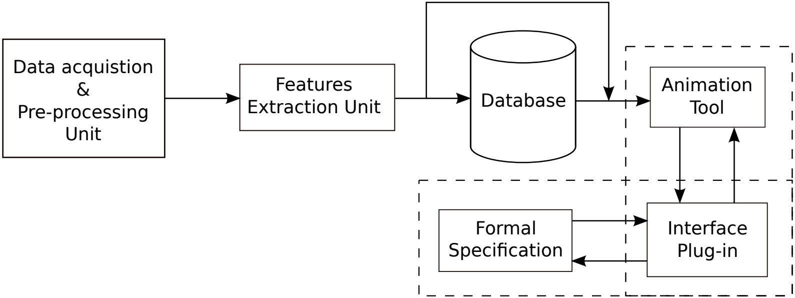
Model	Total number of POs	Automatic Proof	Interactive Proof
Closed-loop model of One-electrode pacemaker			
Abstract Model	304	258(85%)	46(15%)
First Refinement	1015	730(72%)	285(28%)
Second Refinement	72	8(11%)	64(89%)
Third Refinement	153	79(52%)	74(48%)
Closed-loop model of Two-electrode pacemaker			
Abstract Model	291	244(84%)	47(16%)
First Refinement	1039	766(74%)	273(26%)
Second Refinement	53	2(4%)	51(96%)
Third Refinement	122	60(49%)	62(51%)
Total	3049	2147(70%)	902(30%)

Interactive Simulation

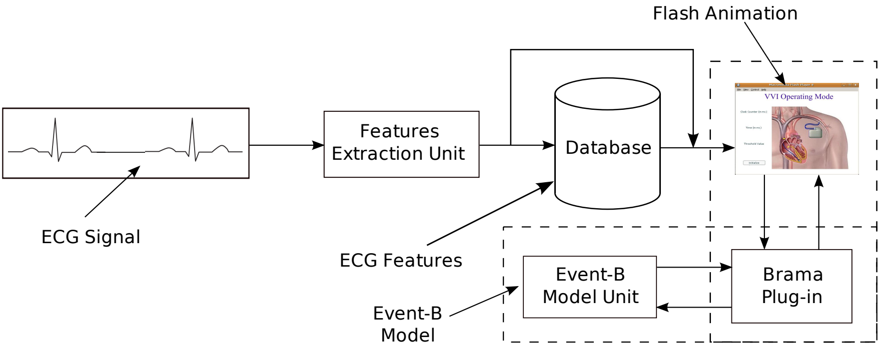
An Architecture of Real-time Animator

Real-Time Animator

Visual representation of a proved formal model using real-time data set to show the system behaviour to domain experts.



Real-time Animation : Pacemaker



ECG Data and Features

- MIT-BIH Database Distribution
- Algorithms to calculate ECG Features
  (http://ecg.mit.edu/index.html)

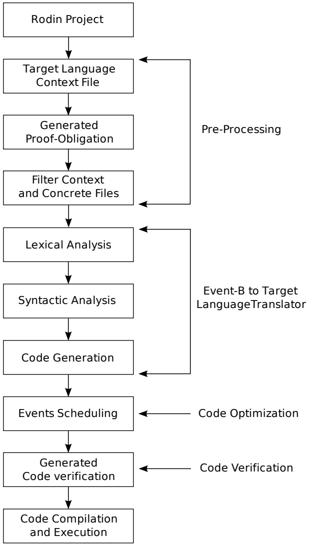
Demo

EB2ALL: Automatic Code Generation from Event-B Models

EB2ALL Code Generation tool

EB2ALL (Event-B to All)

- EB2C
- EB2C++
- EB2J
- EB2C#



EB2ALL Tool Architecture

Pre-Processing

- Objective : Make a system deterministic
- Clean termination approach
- Introduction of new Context Files

Event-B type	Formal Range	C & C++ type	Java type	C# type
tl_int16	$-2^{15}..2^{15}-1$	int	short	short
tl_uint16	$0..2^{16}-1$	unsigned int	-	ushort
tl_int32	$-2^{31}..2^{31}-1$	long int	int	int
tl_uint32	$0..2^{32}-1$	unsigned long int	-	uint
tl_int64	$-2^{63}..2^{63}-1$	-	long	long
tl_uint64	$0..2^{64}-1$	-	-	ulong

Table: Integer bounded data type declaration in different context files

Example :

$Id \in \mathbb{N}_1$ $\qquad$ $Id \in tl\_uint16$

Basic Principles of Code Generation

Event-B	'C' & 'C++' Language	Comment
$n..m$	int	Interger type
$x \in Y$	Y x;	Scaler declaration
$x \in tl\_int16$	int x;	'C' & 'C++' Contexts
$x \in n..m \rightarrow Y$	Y x [m+1];	Array declaration
$x :\in Y$	/* No Action */	Indeterminate Init.
$x :\mid Y$	/* No Action */	Indeterminate Init.
$x = y$	if(x==y) {	Conditional
$x \neq y$	if(x!=y) {	Conditional
$x < y$	if(x<y) {	Conditional
$x \leq y$	if(x<=y) {	Conditional
$x > y$	if(x>y) {	Conditional
$x \geq y$	if(x>=y) {	Conditional
$(x>y) \wedge (x\geq z)$	if ((x>y) && (x>=z) {	Conditional
$(x>y) \vee (x\geq z)$	if ((x>y) || (x>=z) {	Conditional
$x := y + z$	x = y + z;	Arithmetic assignment
$x := y - z$	x = y - z;	Arithmetic assignment
$x := y * z$	x = y * z;	Arithmetic assignment
$x := y \div z$	x = y / z;	Arithmetic assignment
$x := F(y)$	x = F(y);	Function assignment
$a := F(x \mapsto y)$	a = F(x, y);	Function assignment
$x := a(y)$	x = a[y];	Array assignment
$x := y$	x = y;	Scalar action
$a := a \mathbin{<\!\!+} \{x \mapsto y\}$	a[x] = y;	Array action
$a := a \mathbin{<\!\!+} \{x \mapsto y\} \mathbin{<\!\!+} \{i \mapsto j\}$	a[x]=y; a[i]=j;	Array action
$X \Rightarrow Y$	if(!X || Y){	Logical Implication
$X \Leftrightarrow Y$	if((!X || Y) && (!Y || X)){	Logical Equivalence
$\neg x<y$	if(!(x<y)){	Logical not
$x \in \mathbb{N}$	unsigned long int x	Natural numbers
$x \in \mathbb{Z}$	signed long int x	Integer numbers
$\forall$	/* No Action */	Quantifier
$\exists$	/* No Action */	Quantifier

Tool Installation

Web: http://eb2all.loria.fr/(Until Rodin 2.8)

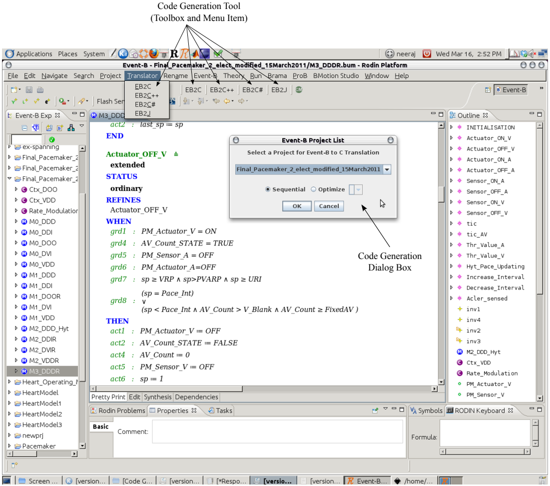


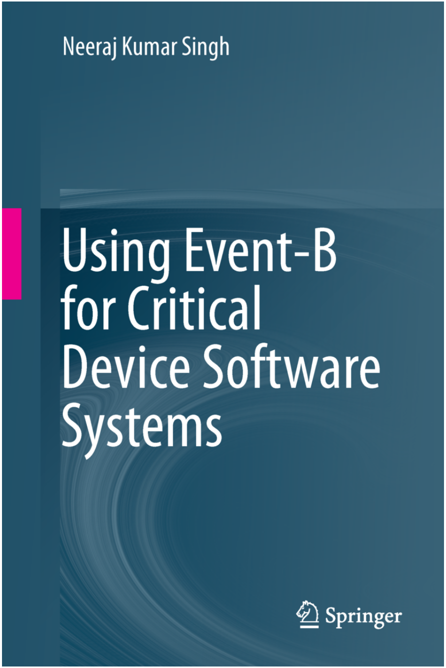
Figure: Screen shots of the Code Generation Tool: EB2ALL

An Example - Code Generation

EVENT Actuator_ON_V
WHEN
Actuator_ON_V.Guard1 : $PM\_Actuator\_V = OFF$
Actuator_ON_V.Guard2 : $(sp = Pace\_Int) \lor (sp < Pace\_Int \land AV\_Count > V\_Blank \land AV\_Count \geq FixedAV)$
Actuator_ON_V.Guard3 : $sp \geq VRP \land sp \geq PVARP$
THEN
Actuator_ON_V.Action1 : $PM\_Actuator\_V := ON$
Actuator_ON_V.Action2 : $last\_sp := sp$
END

```
...
BOOL Actuator\_ON\_V(void)
{
  /* Guards No. 1*/
  if(PM_Actuator_V == OFF){
  /* Guards No. 2*/
  if((sp == Pace_Int) || ((sp < Pace_Int) \&\&
  (AV_Count > V_Blank) && (AV_Count >= FixedAV))){
  /* Guards No. 3*/
  if((sp >= VRP) && (sp >= PVARP) && (sp >= URI)){
  /* Actions */
  PM_Actuator_V = ON;
  last_sp = sp;
  return TRUE;
}}}
  return FALSE;
}
...
```

Demo


Neeraj Kumar Singh
Using Event-B for Critical Device Software Systems
Springer


?
R
D