ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645
Indian Journal of Science and Technology, *Vol 10(20), DOI: 10.17485/ijst/2017/v10i20/104028, May 2017*

Formal Modelling and Verification of the Operational Modes of Pacemaker

Syed Asad Raza Kazmi[1]*, Sana Abubakkar[1], Awais Qasim[1], Syed Hassan Abbas Kazmi[2] and Usman Qamar Qureshi[3]

[1]Department of Computer Science, Government College University, Katchery Road, Lahore – 54000, Pakistan; arkazmi@gcu.edu.pk, sanaa.po@gmail.com, awais@gcu.edu.pk
[2]King Edward Medical University, Nelagumbad, Mayo Hospital Road, Lahore – 54000, Pakistan; hassankaz@kemu.edu.pk
[3]Nishtar Medical College, Multan – 60000, Pakistan; uqqureshi89@gmail.com

Abstract

Objectives: The importance of error free smooth functioning of the heart pacemaker in providing consistent relief to the heart patients has pushed the researchers to devise highly refined bug free devices. Pacemaker is an important safety critical device that is used to keep the heartbeat uniform in patients with low heartbeat. Since the smooth functioning of the pacemaker is responsible to provide oxygen and nutrients to the whole body of the patient in an appropriate ratio, therefore, any conflict in the device would be life threatening. It is therefore extremely necessary to ensure the error free working of such critical health device. **Methods**: Previously, different verification methods have been employed to design and verify safety critical devices with varying degree of reliability and reproducibility. Keeping in view the specifications of the pacemaker, this paper presents a model for the verification and validation of a pacemaker system using the SPIN model checker. **Findings**: The LTL formulas are characterized to handle the uncertainty or any abnormal activity during the process. The system model is designed using SPIN model checker in which different LTL properties are verified. A theoretical description is supported by some experimental results, generated using the existing logics and techniques. **Application**: This work can be further enhanced for the formal verification of critical medical equipment to ensure their correct functioning.

Keywords: Formal Modelling, Pacemaker Modelling, Pacemaker Systems, Temporal Modelling

1. Introduction

Safety critical devices are unique in their functionality whose smooth and error free working is imperative to ensure the safety of property, environment and human life. Safety critical devices are known as life critical devices which work on the methods and tools of safety engineering. These devices make use of the features of FMEA (failure mode and effects analysis) and fault tree analysis based probabilistic risk assessment method[1]. There are many diversified examples of safety critical systems from all spheres of human life including infrastructure, medicine, nuclear engineering, recreation, railway, automotive, aviation and spaceflight hence, safety critical devices have become the integral part of advanced human living style. Therefore, these systems become more and more complex and need more accuracy to function. The careful coding, inspection, documentation, testing, verification

*Author for correspondence

and final analysis of a particular system are the part and parcel of the standard approach to develop conflict free and efficient software for safety critical devices[2–5]. During recent few years medical science has made rapid progress. Computerized automation of medical devices has significantly minimized risks to human life, however, the automation of life saving medical devices need to be conflict and error free[6]. Safety critical devices in health sciences comparatively require more reliability and precision to provide their safe, smooth and error free functioning. Among all the medical safety critical systems, the pacemaker demands more accuracy and bug free implementation because this device is implanted within the human body through surgery[7]. A pacemaker is used to regulate the normal beating of heart[8,9]. This device makes use of electrical impulses produced by the electrodes artificially. The pacemaker is being used in patients naturally having slowed pace-making by heart or in patients with having blockage in the electrical conduction system of heart. The advancement in medical science has made it possible to program the pacemaker externally by cardiologists according to the pace-making requirements of a certain patient[10]. Modern artificial pacemakers perform multiple functions while the most critical function of a pacemaker is to monitor the normal heart beat and native electric rhythm of heart. Pacemaker stimulates the heart ventricles by low voltage and short durational pulse upon non-detection of heart beat within optimum beat-to-beat time period. The pacemakers are designed to work on continuous beat-beat basis. In-addition to its basic function, there are also complicated types of pacemakers, which can stimulate both ventricular and arterial chambers simultaneously under different codes for antibradycardia pacing. The implementation details of most of the safety critical systems/devices are only known to the manufacturer. But, the emergent use of these critical systems in human lives demand their use in more trustworthy way by using highly sophisticated techniques to design, model and verify and validate their software[11,12]. The use of formal methods to ensure the reliability and correctness of critical systems is of utmost importance[11]. Different models have been proposed to formally verify and validate the pacemaker system's specification. Pacemaker system has been evolved over the sequential, concurrent and distributed model using VDM along with C code and Z notation[7]. The model checker PRISM has been used to implement a model-based framework for automated verification of the pacemaker[13]. In this paper, we choose pacemaker system, for which the informal specification has been released in the public domain to model as asynchronous processes with consideration of the time specifications in SPIN. The proposed model is verified safety properties against LTL formulas.

The rest of the paper is organized as follows. In section 2 we describe the mathematical preliminaries needed for the specification of the pacemaker system. In section 3 we explain the working of the pacemaker system along with all the operational modes of the programmable parameters. In section 4 we describe our PROMELA model. In section 5 we show the verification results for a set of desired properties of the model. In section 6 we present the results and section 7 concludes the paper.

2. Preliminaries

Formal Methods are used to define the complex systems by using particular type of mathematical based techniques for the specification of software and hardware systems. Formal methods are authentic and reliable because they make use of formal languages, automata theory, logic and process algebra / calculi and program semantics for verification purpose. Process Algebra is specialized to describe the interactions, communications, and synchronization a groups of independent processes. The process algebra/calculi are a group of related approaches for formal modelling of the concurrent systems. Any system is supposed to show different behaviours according to the requirements. The system's behaviour is the combination of events or actions (which can be refer as a Process) that a system can perform along with other aspects of the system execution such as timing/probabilities. The tools in Process Algebra are algebraic languages for the specification of processes in Calculus of Communicating Systems (CCS), Communicating Sequential Processes (CSP) and LOTOS[14]. The inculcation of concurrency and

timing parameters has been achieved by the extension of CSP to timed CSP, which has ensured the real time functioning of the system. In timed CSP, timing related parameters viz., time out, interruption in timing, wait and so on are being governed by dedicated operators as per user defined information[15].The pacemaker system is a Real-time system and the timing parameters are quite important to describe this system. It has been argued that the formal modelling of real-time systems is necessary to ensure their correct functioning[16].

The following BNF is defined to model the Pacemaker system as real-time concurrent system, where 'e' εå denotes an observable event, A and B range over processes while process parameters is denoted by b. Similarly, integer constant is denoted by i.

A= Stop | Skip – primitives (1)

| action®A – data operation prefixing (2)

| if b then A else B – if-then-else (3)

| A □ B – general choice (4)

| A || B – parallel composition (5)

| A; B – sequential composition (6)

| A / X – hiding (7)

| A ≙ B – process referencing (8)

| Wait[d0; d1] – delay (9)

| A timeout[d] B – timeout (10)

| A interrupt[d] B – timed interrupt (11)

| A within [d0; d1] – react within some time (12)

| A waituntil[d] – wait until (13)

| A deadline[d] – deadline (14)

- Process Stop is only used for halting.
- Process Skip is used for termination.
- Process e ®A engages in event e first and then behaves as A.
- A || B denotes parallel composition of two processes.
- Process A; B behaves as A until A terminates and then behaves as B immediately.
- Process A / X hides the other processes until the A terminates.
- Process A≙B is used for process referencing within the model/programme.

Additionally, it use the Wait, Timeout, Interrupt, Within and Deadline timed process constructs to capture the behavior patterns of the real-time system. By using the BNF, Pacemaker system is modeled over 5 processes i.e. heart, sensor, pace generator, Accelerometer and Timer.

3. Natural Heart and the Pacemaker

According to the different metabolic activities, heart can show a wide range of heartbeat rates. The electrical system of the heart initiates the heartbeat. The Signal starts the contracting of the heart muscle, which produces a

pulse. The sinoatrial node (often abbreviated SA node) is the natural pacemaker of the heart, which is in charge of the initiation of the heartbeat, causes the heart to contract. In order to complete a cardiac cycle the electrical impulse then stimulates atrio-ventricular node (AV), which is relayed to the left and right branches of heart muscles of ventricles. Arrhythmia is the condition, which is the result of electrical disturbance of the heart rhythm being characterized by abnormal heart rhythm with irregular beating pattern. A person can suffer from abnormal slow heartbeat (Bradycardia) and abnormal fast heartbeat (Tachycardia).The pacemaker is being used in patients which having naturally slowed or fast pace-making by heart or in patients with having blockage in the electrical conduction system of heart. A reliable artificial pacemaker system is comprised of three main components including:

- Device (the pace generator or PG)
- Leads
- Device Controller-Monitor (DCM) along with its requisite software.

The characteristics of different pacemakers vary with respect to expected use of the device, working efficiency and robustness to perform in varying environments. In this research, our focus will be on the most critical component of the pacemaker system i.e. the Device (henceforth called Pace generator or PG) to model and validate its components.

4. Operational Modes of the Pacemaker System

The pacemaker system basically works in two modes:

- Permanent mode runs the operations like bradycardia therapy, sensing and pacing pulses.
- Temporary mode tests the pacemaker functionalities and emits reports.

The operating modes are categorized using a code consisting of three or four characters, which are shown in Table1.

- No Response to Sensing (O) means pacing without sensing.
- Triggered Response to Sensing (T) means triggered pacing.
- Inhibited Response to Sensing (I) means during inhibited pacing.
- Tracked Response to Sensing (D) means during tracked pacing.

Table 1. Operating modes of the pacemaker system

	1	2	3	4
Category	**Chambers Paced**	**Chambers Sensed**	**Response To Sense**	**Rate Modulation**
Letters	**O:**None **A:**Atrium **V:**Ventricle **D:**Dual	**O:**None **A:**Atrium **V:**Ventricle **D:**Dual	**O:**None **T:**Triggered **I:**Inhibited **D:**Tracked	**R:**Rate Modulation

While the column IV in the table is optional to represents whether the pacing is rate controlled according to the metabolic activates or not. In this way the Pace Generator (PG) can be programmed over 18 operational modes. These Operational modes specify which chambers of the heart are sensed/paced. Each mode can be written as by using three or four characters code i.e. A (Atria), V(Ventricles),D (both A and V),O (not specified), I (Inhibited), T (Triggered) and R (Rate Controlled).

The 18 modes of the Pacemaker are AAT, VVT, AOO, AAI, VOO, VVI, VDD, DOO, DDI, DDD, AOOR, AAIR, VOOR, VVIR, VDDR, DOOR, DDIR and DDDR.

5. The Pacemaker Model

The behavior of the human heart is captured by the 'heart' process i.e. the contraction event of ventricles and atria. The process 'Sensor' senses the contraction events of the heart and the activity level of rest of the human body. In this way, sensor is work as communicating medium between the heart, the Pulse Generator and the Accelerometer. Accelerometer keeps record of the activities / motion of the human body. It is used to set the response factor (RF), which is used by the 'Rate Controller' to set the appropriate rate of pacing. The 'Pulse Generator(PG)' generates paces on the basis of the sensing data from the sensor. Clock timer process is used to simulate a global clock. An overview of the Pacemaker model is presented in Figure 1.

5.1 Modeling of Process Heart

Heart process is modeled to generate atrial and ventricle contraction events i.e. PaceA and PaceV from time to time. The natural heart can have 6 possible choices to work, which models possible malfunctions. In this dissertation, these 6 choices are modeled as four behaviors of the heart as follows:

Heart() =

/* normal case */

(paceA®Wait[AVD]); (paceV®Wait[AI - AVD]);

Repeat Heart() (15)

/* Ventricle Pace missing */

(paceA® Wait[AVD]) ; (nopace®Wait[AI - AVD]);

Repeat Heart() (16)

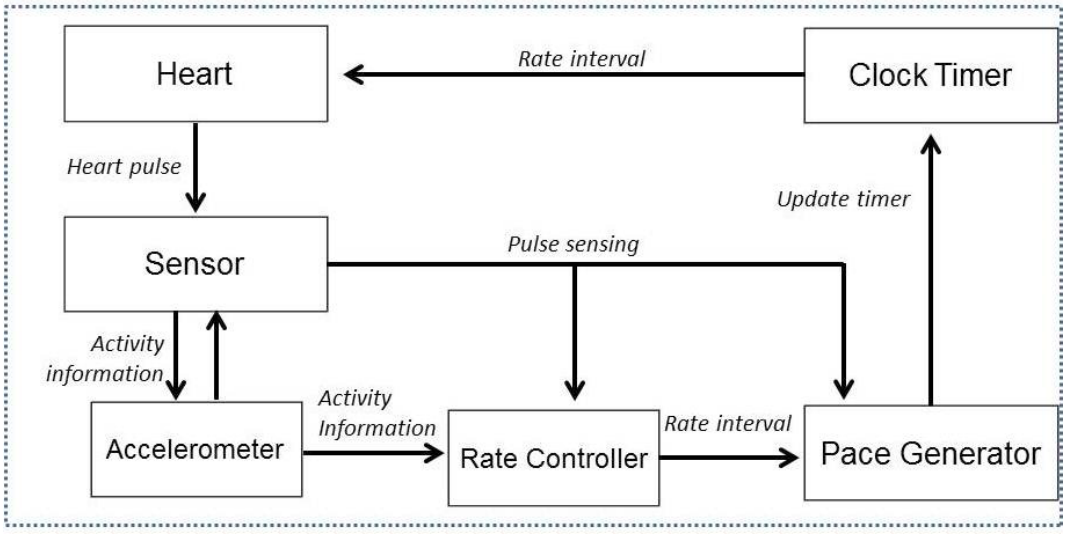

Figure 1. Overview of the Pacemaker model.

/* Atrial Pace missing */

(nopaceA®Wait[AVD]); (paceV®Wait[AI - AVD]); Repeat Heart() (17)

/* Dead Heart */

(nopaceA®Wait[AVD]); (nopaceV®Wait[AI - AVD]); Repeat Heart() (18)

/* Non-deterministic (delayed A or V)*/

(paceA®Wait[AVD+L]); (paceV®Wait[AI - AVD]); Repeat Heart() (19)

(paceA®Wait[AVD]); (paceV®Wait[HI - AVD+L]); Repeat Heart() (20)

Here **AVD** represents the Atrial Ventricular Delay. **AI** denotes interval between two consecutive atrial/ventricular events. **L** represents the random length of time which delays a pulsing event.

5.2 Modeling of Process Sensor

The pacemaker system need to capture the heart pulse and to sense the contraction event of the atria and ventricles in order to keep heat beat steady during non-rate-modulation modes i.e. AAT, VVT, AOO, AAI, VOO, VVI, VDD, DOO and DDI. In the pacemaker system's model, the process 'Sensor' is working to record the paces from the heart and the pace generator. It also modeled to capture the time properties of the contraction event i.e. ARP (Atrial Refractory Period) and VRP (Ventricle Refractory Period) in order to continue the sensing activity with or without trigging pace. An overview of the Sensor working is shown in Figure 2. The process sensor is modeled as 'ASensor' and 'VSensor' that are placed in atria and ventricle respectively.

/* Atria Sensor */

ASensor() = [AS==1] paceA®senseA®Repeat ASensor()

□ [AS==0] paceA®Repeat ASensor() (21)

/* Ventricle Sensor */

VSensor() = [VS==1] paceV®senseV®Repeat VSensor()

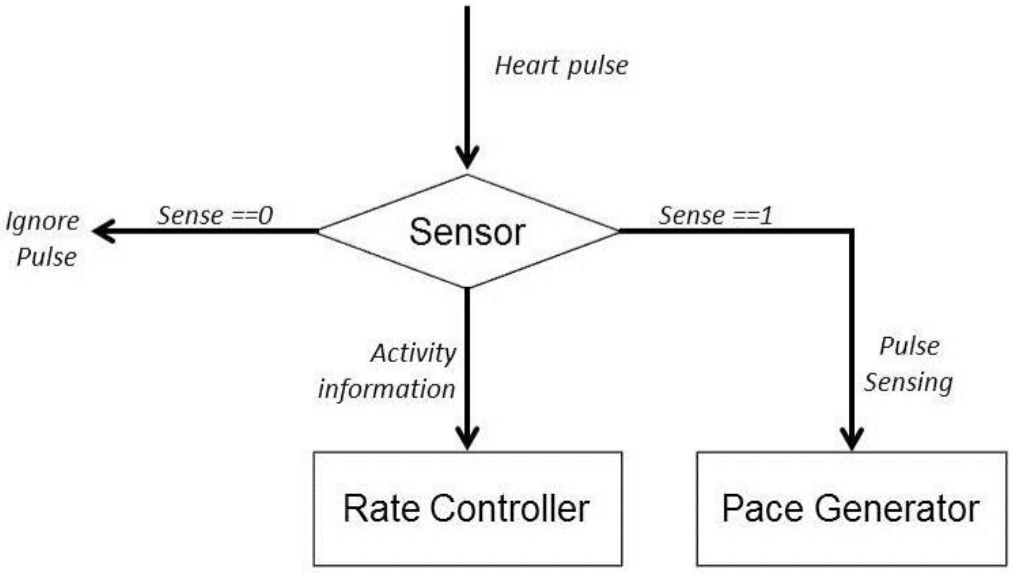

Figure 2. Flow diagram of the process sensor.

□ [VS==0] paceV®Repeat VSensor()

(22)

AS denotes whether an atrial pulse is accepted by the pacemaker or not. Similarly VS denotes whether a ventricle pulse is accepted by the pacemaker or not.

5.3 Modeling of Process Accelerometer and Rate Controller

The pacemaker system needs to keep record of the activity level of the human body but this activity data is only required in rate-modulation modes i.e. AOOR, AAIR, VOOR, VVIR, VDDR, DOOR, DDIR and DDDR. The process 'Accelerometer' is modeled to identify the motion of the human body. The maximum response factor (RF) setting allows the higher incremental change in pacing rate while the minimum response factor (RF) setting allows a smaller change in pacing rate. Figure 3 shows an overview of the process Accelerometer. The process Accelerometer is modeled to capture the activity information from the human body to set the RF parameter as:

/* Accelerometer activity sensor */

AccM() = (ActNone{SAct=ActInfo} ®RateController()

⊓ActNone{SAct=0} ®RateController()

⊓ActVLow{SAct=1} ®RateController()

⊓ActLow{SAct=2} ®RateController()

⊓ActMLow{SAct=3} ®RateController()

⊓ActMedm{SAct=4} ®RateController()

⊓ActMHigh{SAct=5} ®RateController()

⊓ActHigh{SAct=6} ®RateController()

⊓ActVHigh{SAct=7} ®RateController()) within[0];

(23)

SAct is a variable that holds the value of the body activity, which will be used in process Rate Controller to adjust the new rate of the pulse. While within [0] is used to immediately engage the first event of the process once

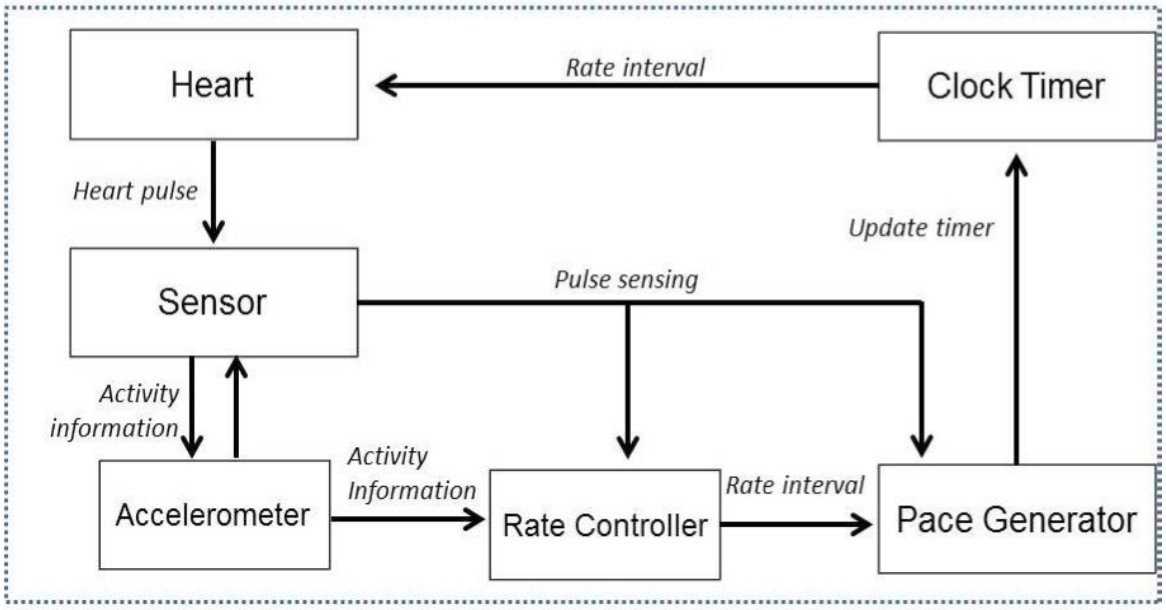

Figure 3. Flow diagram of the process accelerometer.

it is enabled. As the RateController adjusts heart pulse according to the value of sensed activity level by accelerator, the RateController is modeled as:

RateController () = (RateAdjusting {(SAct! = ActInfo) {ActInfo = SAct; NewInterval= URI+(LRI-URI)*(VHigh-SAct)/VHigh;} (24)

/* increaseinterval */

if(Interval <NewInterval) ®(interval=interval+interval*(LRI-MSI)/Recovery Time); (25)

/* decrease interval */

if(Interval >NewInterval) ®(interval=interval-interval*(LRI-MSI)/Reaction Time);

} ®Skip) within[0]; (26)

Here interval is the time between two consecutive pacing events.actInfo denotes the value of the current activity level. NewInterval is the target pulse interval based on the new sensed activity information.

5.4 Modeling of Process Pace Generator

We formally model the Pacemaker system as a process 'Pace Generator'. It is responsible for producing paces in presence or absence of intrinsic beats according to the set parameters. The pacemaker may be programmed to pace either atria or ventricles or both. As discussed in previous section, pacemaker system has 18 different modes in total. We formally modeled only two advanced modes i.e. VVIR and DDDR.

5.4.1 VVIR

In VVIR mode pace generator will only paced and sensed ventricle chamber of the heart. The sensor sensed for the ventricle activity and produces pace if there is no ventricular electrical activity within the specified interval. However, the pace generator inhibits the pace if the ventricle activity performed normally. Figure 4 shows the state machine of VVIR Pace Generator Mode.

PG_{VVIR}= (Heart || VSensor || $pace_{VVIR}$) \ {senseV, paceA, paceV, paceAmissing,

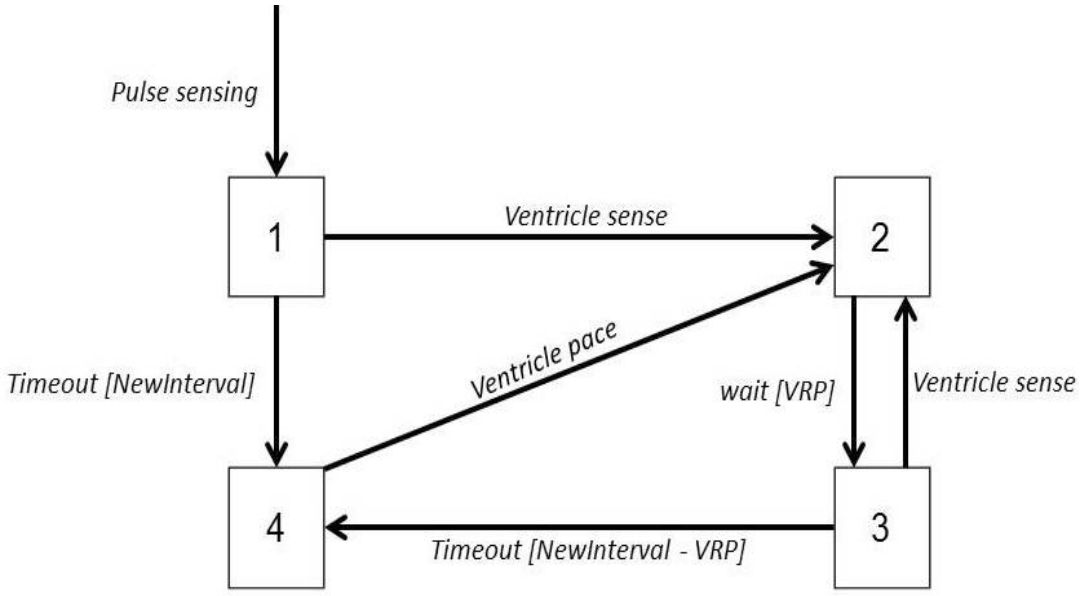

Figure 4. State machine of VVIR pace generator mode.

paceVmissing, VS=1, ActNone, ActVLow, ActLow, ActMLow, ActMedm,ActMHigh, ActHigh, ActVHigh, RateAdjusting}; (27)

$Pace_{VVIR}$() = AccM(); RateController();$Pace_{VVIR2}$();$Pace_{VVIR1}$(); (28)

$Pace_{VVIR1}$()= AccM(); RateController();$Pace_{VVIR3}$(); $Pace_{VVIR1}$(); (29)

$Pace_{VVIR2}$()= ((atomic{senseV®paceV{VS=0} ®Skip}) timeout[interval]

((paceV{VS=0} ®Skip) within[0]); wait[VRP]; (enableVS{VS=1} ®Skip) within[0]); (30)

$Pace_{VVIR3}$()= ((atomic{senseV®paceV{VS=0} ®Skip}) timeout[interval - VRP] ((paceV{VS=0} ®Skip) within[0]); wait[VRP] ; (enableVS{VS=1} ®Skip) within[0]); (31)

5.4.2 DDDR

Pace generator produced atrial pacing if no natural atrial activity happens for set interval and similarly ventricular pacing occurs if no native ventricle activity for set interval following atrial activity. Figure 5 shows the state machine of DDDR Pace Generator Mode.

(insert figure 5 here)

PG_{DDDR} = (setASVS{AS=0; VS=1;} ®Skip) within[0]; (Heart || VSensor || ASensor

|| $Pace_{DDDR}$) \ {senseA, senseV, paceA, paceV, paceAmissing, paceVmissing, AS=1,

VS=1, setASVS, ActNone, ActVLow, ActLow, ActMLow, ActMedm,

ActMHigh, ActHigh, ActVHigh, RateAdjusting}; (32)

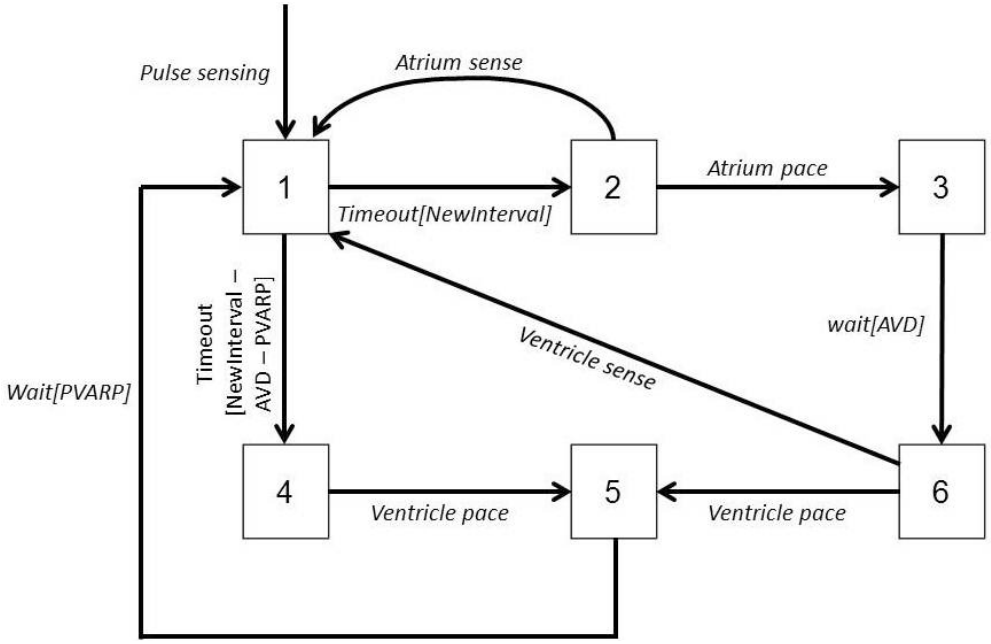

Figure 5. State machine of DDDR pace generator mode.

PG_{DDDR}()= AccM(); RateController();$Pace_{DDDR2}$(); $Pace_{DDDR1}$(); (33)

PG_{DDDR1}()= AccM(); RateController();$Pace_{DDDR3}$(); $Pace_{DDDR1}$(); (34)

PG_{VVIR2}()= (atomic{senseV®paceV{AS=0; VS=0;} ®Skip} timeout[interval]

(paceV{AS=0; VS=0} ®Skip) within[0]); wait[PVARP]; (enableAS{AS=1} ®Skip)

within[0]); (atomic{senseA®paceA{AS=0; VS=1;} ®Skip}

timeout[interval-AVD-PVARP](paceA{AS=0; VS=1} ®Skip) within[0]); (35)

PG_{VVIR3}()= (atomic{senseV®paceV{AS=0; VS=0;} ®Skip} timeout[AVD]

(paceV{AS=0; VS=0} ®Skip) within[0]); wait[PVARP]; (enableAS{AS=1} ®Skip)

within[0]);(atomic{senseApaceA{AS=0; VS=1;} ®Skip}

timeout[interval-AVD-PVARP](paceA{AS=0; VS=1} ®Skip) within[0]); (36)

6. Verification of the Model

It utilizes the SPIN model checker for formal verification of the Pacemaker system. In SPIN the properties of interest have to be specified in PROMELA language and LTL. In this section different LTL properties are defined to verify our model and translated into never claim to run in SPIN.

6.1 AV Delay

The AV delay is a programmable property in pacemaker system. It denotes that the delay between each atrial and ventricle paces always less than a fixed time period. Its corresponding LTL formula is:

AVDelay = G (Sensed AV Delay < Fixed AV Delay) (37)

6.2 Refractory Period

After a paced event in a chamber, the cardiac cell is unable to start another action potential for some duration of time. This duration of time is called Refractory period. Refractory period is a time delay between a sense and a pace in a particular chamber. By this period the time delay between successive paced events of the heart is managed. While model the pacemaker system there are three such properties i.e. Atrial Refractory Period (ARP), Ventricle Refractory Period (VRP) and Post Ventricle Atrial Refractory Period (PVARP). The LTL formulae for the mentioned refectory periods are:

ARP = G ((Last Paced Pulse Atria - Last Sensed Atria) > Atria RP) (38)

VRP = G ((Last Paced Pulse Ventricle - Last Sensed Ventricle) > Ventricle RP) (39)

PVARP = G (((Last Paced Pulse Atria - Last Sensed Atria) > PVARP) &&

((Last Paced Pulse Ventricle - Last Sensed Ventricle) > PVARP)) (40)

6.3 Pace Limit

The pacemaker system must follow a lower and upper range of pulse delivered in one minute to function in accordance with the natural heart. These two ranges are specified as LRL (Lower Rate Limit) and URL (Upper Rate Limit). Its corresponding LTL formula is:

RateLimitA = G(pacing rate of atria < URL && pacing rate of atria > LRL) (41)

RateLimitV = G(pacing rate of ventricle < URL && pacing rate of ventricle > LRL) (42)

6.4 Triggering Property

This property checks if the paces are triggering whenever a sense is detected in a chamber. Its corresponding LTL formula is:

AAT = G (sense Atrial®pace Atria) (43)

VVT = G (sense Ventricle ®pace Ventricle) (44)

6.5 Tracked Property

Tracked property checks if the tracked ventricle pace is supplied after a sense in atria and inhibited if there is a sense in ventricle before that. Its corresponding LTL formula is:

VDD = G (Sense Atria ®F (Pace Ventricle && Sensed AVD < Fixed AVD)) (45)

6.6 Inhibiting Property

This property checks if the pending paces are inhibited in presence of a sense in that chamber. Its corresponding LTL formula is:

AAI = G (sense Atrial®not pace Atria) (46)

VVI = G (sense Ventricle ®not pace Ventricle) (47)

6.7 Rate Limit

It is also necessary to keep the pacing rate less than the maximum sensor rate in rate controlled pacing modes. Its corresponding LTL formula is:

RateLimitA_R = G (Rate of pacing Atrial< Max Sensor Rate) (48)

RateLimitV_R = G (Rate of pacing Ventricle < Max Sensor Rate) (49)

6.8 Rate Control Limit

In rate modulation modes the rate of the pacing must be changed according to the metabolic activity as captured by the accelerometer. This property ensures that the rate of pacing varies according to the activity level of the human body. Its corresponding LTL formula is:

RateControlLimit_A = GF (Rate of pacing A == RF * Activity Threshold) (50)

RateControlLimit_V = GF (Rate of pacing V == RF * Activity Threshold) (51)

7. Conclusion

Our pacemaker system refers to the complete system including the Timed CSP model, LTL properties and the implementation code in SPIN model checker. It models the 18 functional modes of the pacemaker system as a sequential model where the components of the Pulse Generator (PG) are modeled as communicating processes. Major advantage of our work is that it formally modeled the additional functional modes and properties as compared to the previous work done. Our modeling approach utilized the Timed CSP and PROMELA model. The Event-B developed previously only supports invariant properties and has to be extended in order to express handle liveliness. It has successfully verified our model in the SPIN model checker. Formal verification was done for the safety and liveliness properties along with the other 9 desired properties. The researcher provides future directions for the formal verification of safety critical systems to ensure their correct functioning.

8. References

1. Edmund MC, Jeannette MW. Formal methods: State of the art and future directions. ACM Computing Surveys (CSUR). 1996 Dec; 28(4):626–43. CrossRef.
2. Zhihao J, Miroslav P, Rajeev A, Rahul M. Closed-loop verification of medical devices with model abstraction and refinement. International Journal on Software Tools for Technology Transfer. 2014 Apr; 16(2):191–213. CrossRef.
3. Karen S, Lysandra O, Laura M, Robert M. Killed by code: Software transparency in implantable medical devices. Software Freedom Law Center. 2010 Jul; 308-319.
4. Awais Q, Asad RK, Ilyas F. Executable Semantics for the Formal Specification and Verification of E-agents. Indian Journal of Science and Technology. 2015 Jul; 8(16):1–8.
5. Awais Q, Asad RK, Ilyas F. Formal specification and verification of real-time multi-agent system using timed Arc Petri Nets. Advances in Electrical and Computer Engineering. 2015 Jan; 15(3):73–8. CrossRef.
6. Insup L, George JP, Rance C, John H, Bruce HK, Peter L, Harvey R, Lui S. High-confidence medical device software and systems. Computer Application. 2006 Apr; 39(4):33–8. CrossRef.
7. Artur OG, Marcel O. Formal development of a cardiac pacemaker: from specification to code. Brazilian Symposium on Formal Methods, Springer Berlin Heidelberg; 2010. p. 210–25.
8. William HM, Michael OS, William GS, Kristin E, Laurence ME. Recalls and safety alerts involving pacemakers and implantable cardioverter-defibrillator generators. 2001 Aug; 286(7):793–9.
9. Daniel H, Thomas S, Kevin F, Tadayoshi K, William HM. Security and privacy for implantable medical devices. IEEE pervasive computing. 2008 Jan; 7(1):1–7.
10. Àdàm B, Arnold P, Àdàm S, Istvan P. Clinical Observations with Longterm Atrial Pacing. Pacing and clinical electrophysiology. 1998 Jan; 21(1):246–9. CrossRef.
11. Anthony H. Seven myths of formal methods. IEEE software. 1990 Sep; 7(5):11–9. CrossRef.
12. Dominique M, Bernhard S, Alan W. The Pacemaker Challenge: Developing Certifiable Medical Devices. Dagstuhl Reports, Germany: 2014.
13. Lee I, Sokolsky O, Chen S, Hatcliff J, Jee E, Kim B, King A, Mullen-Fortino M, Park S, Roederer A, Venkatasubramanian KK. Challenges and research directions in medical cyber–physical systems. Proceedings of the IEEE. 2012; 100(1):75–90. CrossRef.
14. Jos CMB. A brief history of process algebra. Theoretical Computer Science. 2005 May; 335(2-3):131–46. CrossRef.
15. Steve S. An operational semantics for timed CSP. Information and computation. 1995 Feb; 116(2):193–213. CrossRef.
16. Awais Q, Asad RK. MAPE-K Interfaces for Formal Modeling of Real-Time Self-Adaptive Multi-Agent Systems. IEEE Access. 2016; 4:4946–58. CrossRef.