Hardware-in-the-loop simulation and energy optimization of cardiac pacemakers

Chris Barker[1], Marta Kwiatkowska[2], Alexandru Mereacre[2], Nicola Paoletti[2] and Andrea Patanè[3]

***Abstract*— Implantable cardiac pacemakers are medical devices that can monitor and correct abnormal heart rhythms. To provide the necessary safety assurance for pacemaker software, both testing and verification of the code, as well as testing the entire pacemaker hardware in the loop, is necessary. In this paper, we present a hardware testbed that enables detailed hardware-in-the-loop simulation and energy optimisation of pacemaker algorithms with respect to a heart model. Both the heart and the pacemaker models are encoded in Simulink/Stateflow™ and translated into executable code, with the pacemaker executed directly on the microcontroller. We evaluate the usefulness of the testbed by developing a parameter synthesis algorithm which optimises the timing parameters based on power measurements acquired in real-time. The experiments performed on real measurements successfully demonstrate that the testbed is capable of energy minimisation in real-time and obtains safe pacemaker timing parameters.**

I. Introduction

Implantable cardiac pacemakers are medical devices that use electrical impulses in order to control the heart rate. The pacemaker is implanted in patients who have a slow heart beat (less than 60 BPM) or a block in the electrical conduction system. One of the main challenges is to ensure that pacemakers are safe and consume the least amount of energy possible, since the latter reduces the frequency of re-implantation, thus improving patients' quality of life. Pacemaker safety is typically achieved through developing formal models of the embedded pacemaker software and analysing their correctness using formal methods [4], [11], as well as requiring that the hardware components work within prescribed real-time bounds. Although pacemaker devices include circuits to estimate the battery lifetime, their parameters are typically set by the clinician based only on the heart condition. Energy efficiency can be effectively addressed by designing pacemaker algorithms that optimise energy consumption by regulating the pacing rate for a given patient, taking into account the sensed physiological condition. Design, verification and validation of pacemaker algorithms is a demanding task, particularly in the presence of multiple sensors.

In this paper, we develop a hardware testbed that supports hardware-in-the-loop simulation of model-based pacemaker designs, using hardware circuits that consume small amounts of energy. The purpose of the testbed is to enable detailed evaluation and energy optimisation of pacemaker algorithms in a real hardware setting similar to that used in pacemaker devices, with respect to a model of the heart electrical conduction system. In particular, we focus on minimizing the energy consumption of the pacemaker algorithm based on real measurement data and reproducing different pathologies in the heart model. To this end, we connect the hardware testbed to a power monitor (see Fig. 1). We develop a hardware-in-the-loop (HIL) parameter synthesis algorithm that reads the power measurement data and creates at runtime a surrogate model of the power consumption of the pacemaker. We employ an optimization algorithm that iteratively improves the runtime model and picks the parameter values that guarantee minimal power usage, for a specific heart rhythm. The hardware testbed allows for changing the pacemaker parameters in real-time, as well as retrieving the intrinsic and paced heart beats.

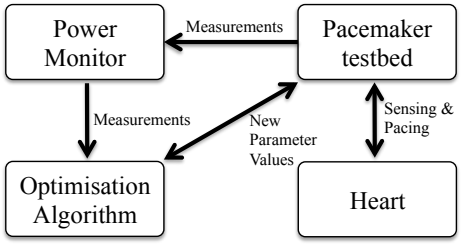


Fig. 1: Parameter synthesis framework

Further, the hardware testbed can be integrated with a variety of sensors to measure patient's physiological condition, including ECG and accelerometer, and can be connected to a sensing circuit that amplifies the signal received from the heart tissue. Although in this work we do not consider personalized parameters, the heart model can be adapted to reproduce the patient's heart rhythm by using signal processing algorithms [10] to synchronise with ECG data.

Related work. In [7], a hardware testbed is developed to simulate the interactions between a formal model of the pacemaker running on a microcontroller and a heart model implemented on a FPGA. Methods for evaluating and optimizing energy consumption in electronic and embedded systems include [2], [5]. Surrogate-based optimization has been used for a number of engineering applications, including hardware-in-the-loop frameworks (see [6], [13]).

II. Methods

A. Hardware testbed

We have implemented the hardware testbed on Arduino Fio, a microcontroller board based on ATmega328P. The

*This work is supported by the ERC AdG VERIWARE and ERC PoC VERIPACE

[1]University of Southampton, UK
[2]Department of Computer Science, University of Oxford, UK
[3]University of Catania, IT

[1,3]Contributed to the work during an internship funded by the ERC PoC Grant VERIPACE

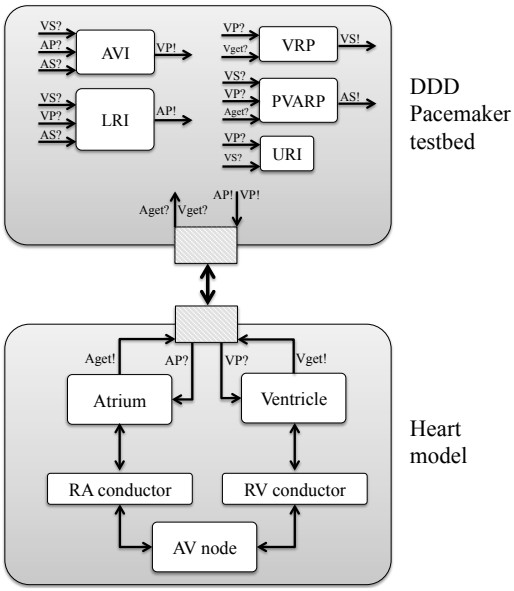


Fig. 2: Architecture of the system

microcontroller runs at 3.3V and 8MHz. It has 2Kb of SRAM, 14 digital input/output pins and 8 analog inputs. It can be connected to a Lithium Polymer battery and has inputs for sensor circuits.

The system consists of two communicating modules and is depicted in Fig. 2. The *pacemaker module* implements the DDD functionality, that is, pacing and sensing of both the atrium and the ventricle. It outputs two pacing signals AP! and VP! for the atrium and ventricle, respectively, and receives two heart signals, Aget? and Vget?, which denote the presence of action potential in the atrium and ventricle. We use serial communication to create a *hardware loop* between the pacemaker and the heart. The *heart module* is implemented as a Simulink™ model on a PC. It has a serial communication component for sending and receiving data from the hardware testbed and a Stateflow™ diagram that implements the electrical conduction system (ECS) of the heart. The heart module receives the AP? and VP? signals from the pacemaker module and sends back the Aget! and Vget! signals.

B. Heart and pacemaker Stateflow™ models

We consider the formal encoding of the heart model of [12] introduced in [1]. The Stateflow™ model consists of five main components, depicted in Fig. 2 (bottom). The *atrium* component models the propagation of the action potential in the atrium, the sino-atrial (SA) node (natural pacemaker of the heart) and the connection between the pacemaker atrial lead and the heart. When the action potential reaches the atrial lead, the atrium component generates the Aget! signal. The atrium component propagates the so-called *antegrade impulse* to the *AV node* through the *RA conductor* component. Intuitively, the RA conductor delays the action potential coming from the atrium. The AV node component acts as a filter for the atrium signals directed to the *ventricle* component through the *RV conductor*, which applies further delays. When the action potential reaches the ventricle lead,

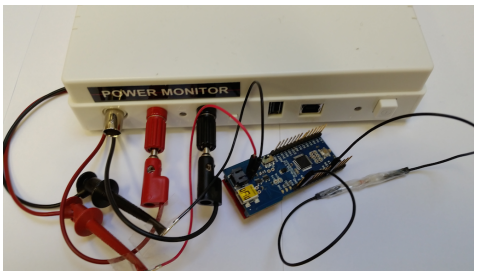


Fig. 3: Power monitoring setup

a Vget! signal is generated. A ventricle beat (either intrinsic or paced) generates a *retrograde impulse*, which is conducted back to the atrium. When an antegrade and a retrograde signal collide in the same conductor, a *fusion beat* occurs and the two signals are annihilated. All five components of the heart model can be instantiated with parameters tailored to individual patients and can reproduce various heart arrhythmias, e.g. bradycardia (slow heart beat), tachycardia (fast heart beat) or AV block.

We have modelled the DDD pacemaker (Fig. 2 top) based on the Boston Scientific specification [3] and [8]. The components are: atrio-ventricular interval (AVI), which maintains synchronisation between the atrium and ventricle; lower rate interval (LRI) and upper rate interval (URI), which set safe bounds for the heart rate; post ventricular atrial refractory period (PVARP) and ventricular refractory period (VRP), which are responsible for sensing, respectively, atrial and ventricular impulses. Every component has associated a timing parameter, which we discuss in Section III.

We use the Matlab™ code generation toolbox to translate the pacemaker and heart models into C code. The pacemaker code is uploaded into the microcontroller, while the heart code is executed on the PC. We enable serial communication between the hardware testbed and the PC to allow sensing and pacing of atrial and ventricular signals.

C. Surrogate-based optimization

We describe the hardware-in-the-loop parameter synthesis framework for finding pacemaker parameters yielding optimal energy consumption values. In Fig. 3 we depict the hardware set-up for power monitoring. We use the Monsoon™ power monitor device to power the pacemaker with 3.7V and record the changes in the energy consumption.

The parameter synthesis algorithm generates randomly a set of parameter values and sends them to the hardware testbed. The hardware testbed and the heart model run for a given amount of time and, at the end of the simulation, the power monitor records the power measurements and sends them back to the synthesis algorithm. This process runs iteratively until the best (near-optimal) parameters are found.

The synthesis algorithm uses a surrogate-driven optimization (SDO) procedure [9]. SDO is a class of simulation-based optimization algorithms that derive a surrogate model of the system under study in order to reduce the number of simulations of the original system. While the original system can be analytically intractable or expensive to simulate, the

surrogate model is fast and amenable to optimization. In our case, an evaluation of the objective function corresponds to a power measurement of the pacemaker device.

SDO algorithms alternate between two main phases: obtain approximate solutions x^* by optimizing the surrogate model; and evaluate the original model at x^*, using the obtained values to improve *at runtime* the accuracy of the surrogate. Our surrogate model is built following the *Kriging* method, which can be seen as a stochastic generalization of linear regression. Given n samples $x^{(1)}, x^{(2)} \ldots, x^{(n)}$, and their respective objective function values $f\left(x^{(1)}\right), f\left(x^{(2)}\right) \ldots, f\left(x^{(n)}\right)$, the method assumes that they are drawn from a model of the form:

$$f\left(x^{(i)}\right) = \vec{g}\left(x^{(i)}\right)^T \cdot \vec{\beta} \; + \epsilon\left(x^{(i)}\right) \quad i = 1, 2, \ldots, n \quad (1)$$

$\vec{g}\left(x^{(i)}\right)^T \cdot \vec{\beta}$ is called the *regression part*, where $\vec{g}\left(x^{(i)}\right)$ is the vector of basis functions and $\vec{\beta}$ is the vector of unknown coefficients estimated through classical regression techniques. ϵ is normally distributed with zero mean and correlation dependent on a weighted Euclidean distance of the n samples. Such weights are the parameters of the surrogate model, and are estimated by maximizing the likelihood function.

The Kriging method is able to provide both an approximate value for $f(x^*)$ and an estimate of the prediction standard error. This property is exploited in the optimization algorithm, where the expected improvement of a point x^* is computed assuming that its objective function value is randomly distributed according to Eq. 1. We remark that this method provides an *approximate optimal solution* and a *well-behaved approximation of the energy consumption function*. Indeed, this is accurate in the region of interest (i.e. the optimal region of the input space), besides being cheap to evaluate and analyse.

III. Results

We synthesize the pacemaker parameters that minimize the total electric current during 10 seconds of HIL simulation. Along with the optimal parameters, we also derive a surrogate statistical model (see Sect. II-C) that, given the pacemaker parameters, estimates at runtime the flow of the current. We consider two different heart conditions: *bradycardia*, i.e. slow heart rate, which we reproduce by decreasing the firing rate of the SA node in the heart model; and a *Wenckebach AV block*, which causes the loss of ventricular beats due to the progressive prolongation of the AV conduction time. This is simulated by setting in the heart model a high threshold voltage of the AV node, which increases its depolarisation time.

We consider the following pacemaker parameters: TLRI (default value 1000 ms), which sets a lower bound on the heart rate; TAVI (default, 150 ms), which mimics the time needed for an atrial impulse to reach the ventricle; TURI (default, 500 ms), which sets an upper bound for the heart rate, i.e. the time before pacing the ventricle, after an atrial stimulus has occurred and TAVI elapsed; and TPVARP (default, 250 ms), that is, the time during which atrial sensing is disabled after a ventricle event, so as to filter out atrial noise. Since we aim at synthesizing parameters that also provide a safe heart rhythm, we add a penalty to the objective function for parameters yielding a heart rate outside the range $[60, 120]$ BPM.

Figure 4 summarizes the results of energy optimization. In the AV block case (Fig. 4a), the minimum current is achieved at $\mathrm{TLRI} = 2047$ ms and $\mathrm{TAVI} = 333$ ms, even if other regions with close power consumption values can be observed. These values overestimate the default ones but are able to ensure a correct heart rhythm because, with an AV block, the heart possesses an adequate atrial frequency. This means that the sensed atrial impulses always precede the time-out for atrial pacing. On the other hand, overpacing occurs for low TLRI and TAVI, thus violating the heart safety property and yielding high objective function values (red region in the bottom-left corner of Fig. 4a). We also evaluate the standard deviation (SD) associated with the surrogate Kriging model (Fig. 4d). This analysis is of crucial importance because, ideally, pacemaker parameters that yield low consumption with low variability are preferred. The variability tends to decrease close to sampled points, even if they can be surrounded by regions with high SD. This is the case for the obtained optimal solution. On the other hand, parameter regions with both low current flow and SD can be found approximately for $\mathrm{TLRI} \in [600, 1000]$ ms and $\mathrm{TAVI} < 1000$ ms, or for $\mathrm{TLRI} < 1000$ ms and $\mathrm{TAVI} \in [800, 1000]$ ms.

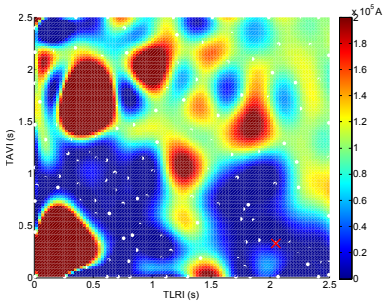
In the bradycardia case (Fig. 4b and 4e), the minimum total current is 1630 A for $\mathrm{TLRI} = 1276$ ms and $\mathrm{TAVI} = 471$ ms. In general, such parameter values are not recommended for patients with slow SA firing rate, since $\mathrm{TLRI} = 1276$ ms implies a pacing rate in the atrium of 47 BPM. Thus, the combination of intrinsic heart impulses, albeit slow, with the paced ones ensures the required safe heart rate. In this case, the region approximately given by $\mathrm{TLRI} \in [500, 1000]$ and $\mathrm{TAVI} \in [0, 500]$ yields both low current values and low SD.

In Fig. 4b and 4e, we estimate the total current without penalizing unsafe heart parameters. The optimal parameter values are $\mathrm{TURI} = 1899$ ms and $\mathrm{TPVARP} = 1211$ ms, but we observe that low current values can be achieved for approximately $\mathrm{TURI} > 1300$ ms. Clearly, high TURI values imply a lower pacing rate and thus lower energy consumption, even if such parameters cannot provide an appropriate heart rate with an AV block condition. On the other hand, the value of TPVARP does not significantly affect the total current.

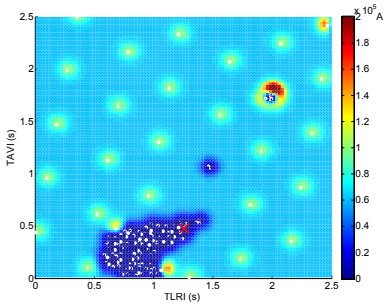
We report that the SDO algorithm is able to synthesize pacemaker parameters that improve the energy consumption obtained with the default parameters. Moreover, it ensures fast convergence to the optimal value, as shown in Fig. 5.

IV. Conclusion and future work

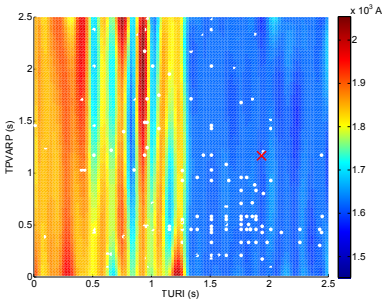
We developed a hardware testbed to facilitate hardware-in-the-loop simulation and energy optimisation of implantable cardiac pacemakers. We implemented the pacemaker and



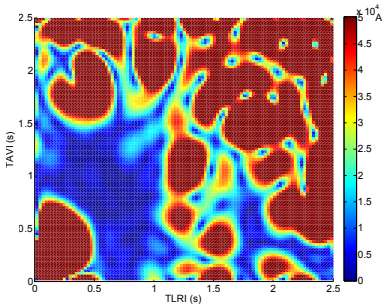
(a) μ, AV block. Min current: 1618 A, at TLRI $=$ 2047 ms and TAVI $=$ 333 ms. Improvement: 1.64%.



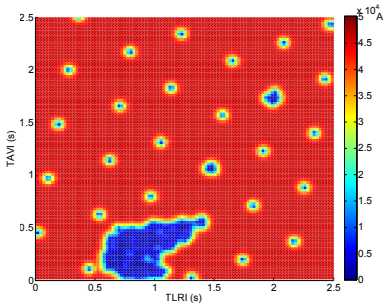
(b) μ, bradycardia. Min current: 1630 A, at TLRI $=$ 1276 ms and TAVI $=$ 471 ms. Improvement: 0.85%.



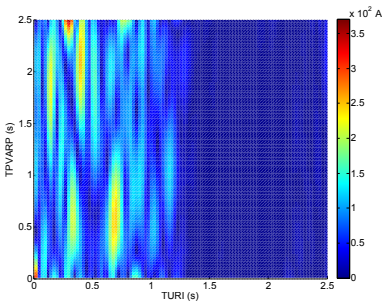
(c) μ. Min current: 1615 A, at TURI $=$ 1899 ms and TPVARP $=$ 1211 ms. Improvement: 10.6%.



(d) σ for plot (a).



(e) σ for plot (b).



(f) σ for plot (c).

Fig. 4: Statistical estimation and optimization of the pacemaker total electrical current in 10 seconds simulation. Parameters are TLRI and TAVI for plots a,b,d,e; and TURI and TPVARP for plots c,f. Expected values are in top plots, standard deviations in the bottom plots. Results are obtained through SDO, with 30 initial samples. Optimal parameters are indicated with a red cross. For each experiment, 150 unique points are sampled (white dots). In plots a,b,d,e, we add penalty of 10^5 A to points yielding a heart rate not in $[60, 120]$ BPM. The relative improvement with respect to the expected total electrical current for the default parameters is reported. Parameter ranges are $[10, 2500]$ ms.

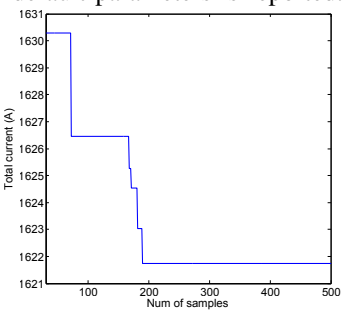


Fig. 5: Best total current value at increasing number of samples. The experiment is as in Fig. 4b.

heart models as hybrid automata in Simulink/Stateflow, and encoded the pacemaker component into hardware. We demonstrated the usefulness of our testbed on the synthesis of safe and energy-efficient pacemaker parameters, by integrating power measurements in the optimization loop and deriving predictive models of energy consumption. As future work, we plan to generalize our approach to directly optimize pacemaker code.

References

[1] B. Barbot, M. Kwiatkowska, A. Mereacre, and N. Paoletti. Estimation and verification of hybrid heart models for personalised medical and wearable devices. In *CMSB. To appear*, 2015.

[2] L. Benini and G. d. Micheli. System-level power optimization: techniques and tools. *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, 5(2):115–192, 2000.

[3] Boston Scientific. Pacemaker system specification. 2007.

[4] T. Chen, M. Diciolla, M. Kwiatkowska, and A. Mereacre. Quantitative verification of implantable cardiac pacemakers over hybrid heart models. *Information and Computation*, 236:87–101, 2014.

[5] J. Flinn and M. Satyanarayanan. Powerscope: A tool for profiling the energy usage of mobile applications. In *Mobile Computing Systems and Applications, 1999. Proceedings. WMCSA'99. Second IEEE Workshop on*, pages 2–10. IEEE, 1999.

[6] T. Hemker, H. Sakamoto, M. Stelzer, and O. von Stryk. Hardware-in-the-loop optimization of the walking speed of a humanoid robot. In *Proc. of CLAWAR*, 2006.

[7] Z. Jiang, M. Pajic, and R. Mangharam. Cyber–physical modeling of implantable cardiac medical devices. *Proceedings of the IEEE*, 100(1):122–137, 2012.

[8] Z. Jiang, M. Pajic, S. Moarref, R. Alur, and R. Mangharam. Modeling and verification of a dual chamber implantable pacemaker. In *TACAS*, volume 7214 of *LNCS*, pages 188–203. Springer, 2012.

[9] S. Koziel and X.-S. Yang. *Computational optimization, methods and algorithms*, volume 356. Springer, 2011.

[10] M. Kwiatkowska, H. Lea-Banks, A. Mereacre, and N. Paoletti. Formal modelling and validation of rate-adaptive pacemakers. In *IEEE International Conference on Healthcare Informatics (ICHI)*, pages 23–32. IEEE, 2014.

[11] M. Kwiatkowska, A. Mereacre, and N. Paoletti. On quantitative software quality assurance methodologies for cardiac pacemakers. In *ISoLA*, LNCS, pages 365–384. Springer, 2014.

[12] J. Lian, D. Mussig, and V. Lang. Computer modeling of ventricular rhythm during atrial fibrillation and ventricular pacing. *Biomedical Engineering, IEEE Transactions on*, 53(8):1512–1520, 2006.

[13] M. Mehari, E. De Poorter, I. Couckuyt, D. Deschrijver, J. Gerwen, T. Dhaene, and I. Moerman. Efficient multi-objective optimization of wireless network problems on wireless testbeds. In *Network and Service Management (CNSM), 2014 10th International Conference on*, pages 212–217. IEEE, 2014.