VLSI Implementation of the Pacemaker Unit of the Interstitial Cell of Cajal

Ganapathy Subramanium Sundar
Department of Bioengineering
University of California, San Diego
San Diego, CA-92093

Abstract

The digestion of the food in the gastro-intestinal tract is aided by the peristaltic contractions of the gastric muscles. This gastric motility is a result of the spontaneous rhythmic pacemaker activity produced by the interstitial cells of Cajal (ICC), which line the gastro-enteric system. These cells generate periodic action potentials termed as slow waves, which are responsible for gastric motility. The objective of this project is to implement a VLSI model of a simple ICC network capable of producing slow waves generated by the ICC network of the stomach. The cells of the network are designed using a mathematical model where the membrane voltage of the ICC is based on the flow of Ca\(^{2+}\) ions and Na\(^+\) ions.

1 Introduction:

The human digestive system has the tasks of ingestion, digestion, absorption and assimilation of food. The food taken in is pushed through the alimentary canal by a series of rhythmic contractions of the smooth muscles lining the gastro-intestinal (GI) tract called peristalsis. These contractions are caused by a set of electrical-sIGNALS called "slow waves". The slow waves, in turn, are generated by the Interstitial cells of Cajal which also line the GI tract. The slow waves are a summation of a large number of various functional cell membrane fluctuations called "Unitary Potentials (UP)". The unitary potentials are generated from a group of cell organelles called "Pacemaker Unit (PU)". The UP is generated by the flow of Ca\(^{2+}\) ions through the Pacemaker Unit.

Abnormalities in the Interstitial Cells of Cajal could result in gastric diseases like Irritable Bowel Syndrome and Pylorus Dysfunction. In this project, a VLSI model of the Pacemaker Unit, which is the most fundamental unit of the ICC, is built using VHDL from a proven biophysical model.

2 Biophysical Background:

2.1 The Interstitial Cell of Cajal:

The Interstitial Cells of Cajal were first isolated by Santiago Ramon y Cajal. These cells are found along the entire gut wall including the myenteric-plexus and at the interface between the circular muscle and the submucosa. These cells are neither neurons nor muscular cells
and are supposed to be derived from fibroblasts. The ICC are electrically coupled to the enteric muscle cells through gap junctions. The slow waves in the digestive system are generated by a large group of these cells.

Each part of the digestive system has its own characteristic slow-wave. For example, in the human digestive system, the frequency of the slow-waves is 3 cycles per minute in the stomach and 12 cycles per minute in the duodenum. This variation is largely attributed to the difference in the ICC population in the various digestive organs. Figure 1 shows the different population of the ICC in the rat stomach and the rat intestine.

![Figure 1: ICC distribution along the murine stomach and murine intestine](image)

### 2.2 The Pacemaker Unit:

[2] suggests that the Interstitial Cell of Cajal is fundamentally made up of a number of small functional units which are responsible for the generation of the slow waves. These units are called the pacemaker units and essentially are made up of four components:

#### 2.2.1 The Endoplasmic Reticulum (ER):

It is the intracellular Ca$^{2+}$ store that cycles Ca$^{2+}$ by release via the IP$_3$R and is sequestered back again by the sarcoplasmic reticulum calcium ATPase (SERCA) pump proteins.

#### 2.2.2 The Mitochondrial Subspace (MT):

It regulates the cellular ATP production through the mitochondrial calcium uniporter (MCU) and the Mitochondrial Sodium/Calcium Exchanger (NCX).

#### 2.2.3 Cytosolic Subspace ($S_1$ and $S_2$):

These two form the remaining of the pacemaker unit and includes the Non-Selective Cation conductance exchanger (NSCC) and the intracellular Ca$^{2+}$ fluxes ($S_1S_2$).

### 2.3 The Unitary Potential:

The Unitary Potential is generated in the Pacemaker Unit as a consequence of the flow of Ca$^{2+}$ ions. Initially Ca$^{2+}$ ions flow into the $S_1$ subspace which then diffuse to $S_2$. Sufficient Ca$^{2+}$ entry into $S_2$ raises the IP$_3$ R open probability causing a flow of Ca$^{2+}$ from the Endoplasmic Reticulum. The mitochondria also gets activated through the MCU channel, causing rapid mitochondrial Ca$^{2+}$ accumulation. To maintain ionic concentrations in the ER, the ER takes up Ca$^{2+}$ ions through the SERCA pump. The deficiency in the ions in $S_1$ causes the NSCC channel to open up taking up both Na$^+$ and Ca$^{2+}$ ions. This depolarization is then followed by a repolarization where mitochondria releases the excess...
Ca$^{2+}$ ions through NCX and $S_1$ releases Na$^+$ ions into the extracellular space. This process is cyclic and gives rise to a unitary potential cycle.

Figure 2: The schematic diagram of the Pacemaker Unit and the flow of ions.

2.4 The Model Framework:

[2] models the Unitary Potential as a state model of 5 differential equations governing the basic parameters. The state variables are:

- $V_m$: Membrane Potential
- $C_{S1}$: Ca$^{2+}$ concentration in $S_1$
- $C_{S2}$: Ca$^{2+}$ concentration in $S_2$
- $C_{ER}$: Ca$^{2+}$ concentration in ER
- $C_{MT}$: Ca$^{2+}$ concentration in MT
- $N_{S1}$: Na$^+$ concentration in $S_1$
- $H$: opening gate variable
- $\Phi_3$: IP$_3$R slow variable

The state equations are given by:

\[
\frac{dV_m}{dt} = -\frac{1}{C_m}(I_{Na} + I_{Ca})
\]

\[
\frac{dC_{S1}}{dt} = J_{SIS1} + \lambda_{MTS1} J_{NCX} - \left( \frac{\delta_{[IP]}(V)}{V_{slo}} \right) J_{in} - \lambda_{ES1} J_{SERCA}
\]

\[
\frac{dC_{S2}}{dt} = J_{SICY} + \lambda_{ERTS1} J_{IPR} - \lambda_{S1} J_{SIS2} - \lambda_{ES2} J_{MCU}
\]

\[
\frac{dC_{ER}}{dt} = J_{SERCA} - J_{IPR}
\]

\[
\frac{dC_{MT}}{dt} = f_s J_{MCU} - J_{NCX}
\]

\[
\frac{dN_{S1}}{dt} = \left( \frac{\delta_{[IP]}(V)}{V_{slo} Z_{Na}} \right) J_{Na}
\]

\[
\frac{dH}{dt} = \delta_1 (1 - H) \left( \frac{P \phi_1}{P \phi_1 + 1} \right) H
\]
Where the $J_i$ is the $Ca^{2+}$ flux due to the particular channel $i$, $\varphi_1$ and $\varphi_2$ are the opening and closing gate variables of $IP_3R$ and $f_m$ is the mitochondrial buffering rate.

3. System Implementation:

3.1 Basic Block Diagram:

The Pacemaker Unit's digital VLSI model has two inputs viz. the Clock and the Reset and a 32 bit output for the output voltage generated. This is as shown in Figure 3.

![Figure 3: Black Box of Pacemaker Unit](image)

The Pacemaker Unit's block diagram is shown in the following diagram. The constants and the parameters are stored in a memory block in the 32 bit IEEE-754 format. The main advantage of this format is that floating point numbers are easy to represent and compute. The membrane voltage and the other state parameters are calculated in the computation block. The system is a positive edge triggered system and consists of 32 bit floating point multipliers, adders and subtractors.

To verify the accuracy of the model, the system was first simulated on MATLAB with an initial resting potential of $-70.1$ mV. The system was also simulated in MATLAB to determine the clock frequency required to run the system so as to obtain a unitary potential of 3 cycles/minute as the case with most mammals.

The actual model was simulated using VHDL on Altera’s FPGA software Quartus II. The output waveforms were analyzed using ModelSim and the Register Transfer Level (RTL) schematic was obtained.
4. Results:

Figure 5 shows the MATLAB simulation of the system using MATLAB’s ode23 solver. By a hit-and-trial method, the system was found to generate a unitary potential of the desired frequency when the samples were spaced at 1.54 ms. Thus it was decided that the clock frequency of the system to be at 6.4 kHz. Figure 6 shows the various Ca^{2+} concentrations across the different parts of the Pacemaker Unit.

Figure 5: MATLAB: Unitary Potentials (Amplitude 3 mV pk-pk)

Figure 6: MATLAB Ca^{2+} ion flow in various components of the Pacemaker Unit
The system was built using the behavioural model and employed the Euler's method for solving the differential equations. Appendix A shows the RTL schematic synthesized for the Pacemaker Unit.

5. Conclusion and Future Work:

Thus the VLSI model of the Pacemaker Unit was obtained using VHDL and verified with MATLAB simulations. The Interstitial Cell of Cajal has more than one Pacemaker Unit. Hence the collective summation of all the Pacemaker units has to be studied. Also the gastric wall is lined with more than one ICC. Hence it is necessary to study the effects of many ICCs and the effect of the ICC population on the characteristics of the slow wave generated.

References
Appendix A

RTL Schematic of the Pacemaker Unit
Appendix B

VHDL Code for the Pacemaker Unit

library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_arith.all;
use IEEE.math_real.all;
--library IEEE_PROPOSED;
--use IEEE_PROPOSED.fixed_float_types.all;
--use IEEE_PROPOSED.fixed_pkg.all;
--use IEEE_PROPOSED.float_pkg.all;

entity Pace_Unit is
  port (clk,rst:in STD_LOGIC;V_out:out STD_LOGIC_VECTOR(31 downto 0));
end Pace_Unit;

architecture PU of Pace_Unit is

-- UP Model Parameters
constant g_Ca : real := 0.01;
constant E_NSCC : real := 0.0;
constant g_capNSCC_Ca : real := 0.12;
constant g_capNSCC_Na : real := 220.0;
constant K_NSCC : real := 0.12;
constant h_NSCC : real := 3.0;
constant g_PM : real := 420.0;
constant K_PM : real := 1.0;
constant g_Na : real := 15000.0;
constant K_Na : real := 10000.0;
constant h_Na : real := 4.0;
constant V_SERCA : real := 100000.0;
constant A_2 : real := 0.0006;
constant A_4 : real := 3.57;
constant A_5 : real := 0.000027;
constant A_6 : real := 0.0000231;
constant V_MCU : real := 800.0;
constant K_MCU : real := 10.0;
constant K_INH : real := 10.0;
constant h_INH : real := 4.0;
constant V_NCX : real := 0.5;
constant K_NCX : real := 0.3;
constant u_S1S2 : real := 0.04;
constant k_IpR : real := 2000.0;
constant k_1 : real := 0.0;
constant k_min1 : real := 6.4;
constant k_2 : real := 4.0;
constant r_2 : real := 200.0;
constant r_min2 : real := 0.0;
constant r_4 : real := 750.0;
constant R_1 : real := 36.0;
constant R_3 : real := 300.0;
constant g_alpha : real := 0.02;
constant g_beta : real := 300.0;
constant K_beta : real := 2.0;
constant h_beta : real := 2.0;
constant K_m : real := 0.01;
constant B_m : real := 100.0;
constant y_S1 :real := 100.0;
constant y_S2 :real := 1.0;
constant y_ER :real := 20.0;
constant y_MT :real := 200.0;
d_S :real := 26.0;
constant C_m :real := 20.0;
constant Z_Ca :real := 2.0;
constant Z_Na :real := 1.0;
constant V_t :real := 26.7;
constant C_O :real := 1800.0;
P :real := 1.0;
signal V_m :real := -70.1;
signal C_S1 :real := 0.12;
signal C_S2 :real := 0.023;
signal C_ER :real := 203.0;
signal C_MT :real := 0.22;
signal N_S1 :real := 10000.0;
signal H : real := 0.0;
signal phi_3 :real := 0.306;
signal a_phi3,b_phi3,phi_1,phi_min1,phi_2,J_IPR,J_S1S2,J_NCX,J_MCU,J_SERCA : real := 0.0;
signal I_Na,I_PM,g_NSCC_Ca,g_NSCC_Na,I_NSCC_Ca,I_NSCC_Na,E_Ca,I_Ca,I_iCa,I_iNa,f_m:
real := 0.0;
signal l_MTS1,l_ERS1,l_ERS2,l_S1S2,l_MTS2 : real := 0.0;
begin
process(clk,rst)
begin
  if rst = '1' and rising_edge(clk) then
    V_out <= CONV_STD_LOGIC_VECTOR(INTEGER(V_m),32);
    a_phi3 <= g_alpha;
    b_phi3 <= g_beta*((C_S2**h_beta)/(K_beta**h_beta+C_S2**h_beta));
    phi_3 <= phi_3 + (a_phi3 - b_phi3*phi_3);
  phi_1 <= (k_1*R_1+r_2*C_S2)/(R_1+C_S2);
  phi_min1 <= ((k_min1+r_min2)*R_3)/(R_3+C_S2);
  phi_2 <= (k_2*R_3+r_4*C_S2)/(R_3+C_S2);
  J_IPR <= k_IPR*(((P*phi_1*H)/(P*phi_1+phi_min1))**4.0)*(C_ER-C_S2);
  J_S1S2 <= u_S1S2*(C_S2-C_S1);
  J_NCX <= V_NCX*(C_MT/(K_NCX+C_MT));
  J_MCU <= V_MCU*(C_S2**2.0)/(K_MCU**2.0+C_S2**2.0);
  J_SERCA <= V_SERCA*(C_S1-A_2*C_ER)/(1.0+A_4*C_S1+A_5*C_ER+A_6*C_S1*C_ER);
  l_MTS1 <= y_MT/y_S1;
  l_ERS1 <= y_ER/y_S1;
  l_ERS2 <= y_ER/y_S2;
  l_S1S2 <= y_S1/y_S2;
end
end

l_MTS2 <= y_MT/y_S2;
f_m <= 1.0/(1.0+(K_m*B_m/(K_m+C_MT**2.0)));
I_iNa <= I_NSCC_Na+I_Na;
I_iCa <= I_NSCC_Ca+I_Ca;
V_m <= V_m - ((I_iCa+I_iNa)/C_m);
C_S1 <= C_S1 + (J_S1S2+l_MTS1*J_MCU);
C_S2 <= C_S2 + (J_S1S2+I_MTS1*J_NCX);
C_ER <= C_ER + (J_SERCA-J_IPR);
C_MT <= C_MT + f_m*(J_MCU-J_NCX);
N_S1 <= N_S1 - ((d_S/Z_Na)*I_iNa);
H <= H+ ((phi_3*(1.0-H)) - (((P*phi_1*phi_2)/(P*phi_1+phi_min1)*H)));
-V_out <= CONV_STD_LOGIC_VECTOR(INTEGER(V_m),32);
end if;
end process;
end PU;