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 sponta- neous rhythmic pacemaker activity produced by the interstitial cells of Cajal(ICC), which line the gastro-enteric system. The interstitial cells of Cajal are a special- ized group of neuro-muscular cells and are derived from fibroblasts. 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 mathemat- ical 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)l 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 charactersitic 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.

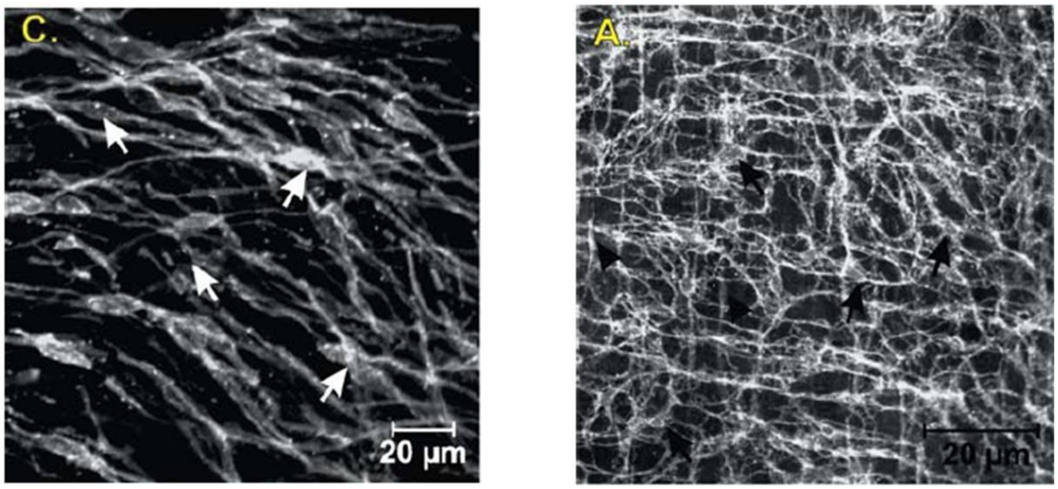


Figure 1: ICC distribution along the murine stomach and murine intestine[2]

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_3R and is sequestered back again by the sarcoeondoplasmic reticulum calcium ATPase (SERCA) pump proteins.

2.2.2 The Mitochondrial Subspace (MT) :

It regulates the cellular ATP production through the mitochondrial calcium uni-porter(MCU) and the Mitrochondrial Sodium/Calcium Exchanger(NCX).

2.2.3 Cytosolic Subspace(S_1 and S_2):

These two form the reamining of the pacemaker unit and includes the Non-Selective Cation conductance exchanger(NSCC) and the intracellular Ca^{2+} fluxes(S1S2).

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 though 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.

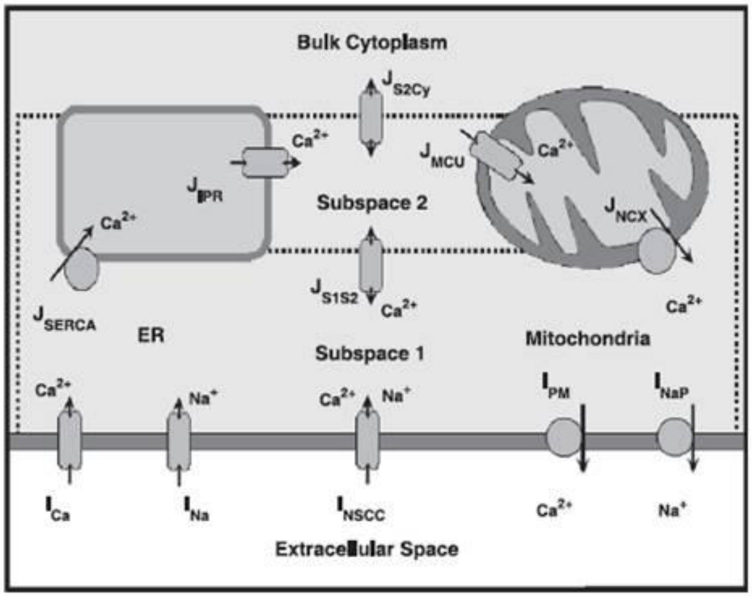


Figure 2: The schematic diagram of the Pacemaker Unit and the flow of ions.

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
Φ_3: IP_3R slow variable

The state equations are given by:

$$\frac{dV_m}{dt} = -\frac{1}{C_m}(I_{iNa} + I_{iCa})$$

$$\frac{dC_{S1}}{dt} = J_{S1S2} + \lambda_{MT/S_1} J_{NCX} - \left(\frac{\delta_{S(PU)}}{V_{Scale} Z_{Ca}}\right) I_{iCa} - \lambda_{ER/S_1} J_{SERCA}$$

$$\frac{dC_{S2}}{dt} = J_{S2Cy} + \lambda_{ER/S_2} J_{IPR} - \lambda_{S_1/S_2} J_{S1S2} - \lambda_{MT/S_2} J_{MCU}$$

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

$$\frac{dC_{MT}}{dt} = f_m (J_{MCU} - J_{NCX})$$

$$\frac{dN_{S1}}{dt} = -\left(\frac{\delta_{S(PU)}}{V_{Scale} Z_{Na}}\right) I_{iNa}$$

$$\frac{dH}{dt} = \phi_3 (1 - H) - \left(\frac{P\phi_1\phi_2}{P\phi_1 + \phi_{-1}}\right) H$$

Where the J_i is the Ca^{2+} flux due to the particular channel i, φ_1 and φ_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

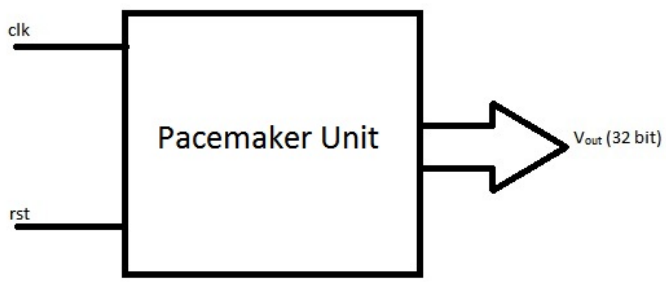


Figure 3: Black Box of Pacemaker Unit

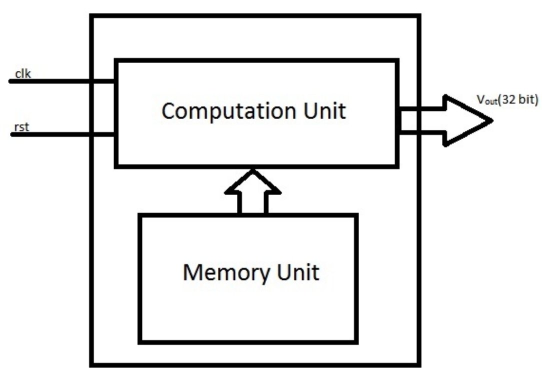


Figure 4: Block Diagram of the Pacemaker Unit

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 a 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.

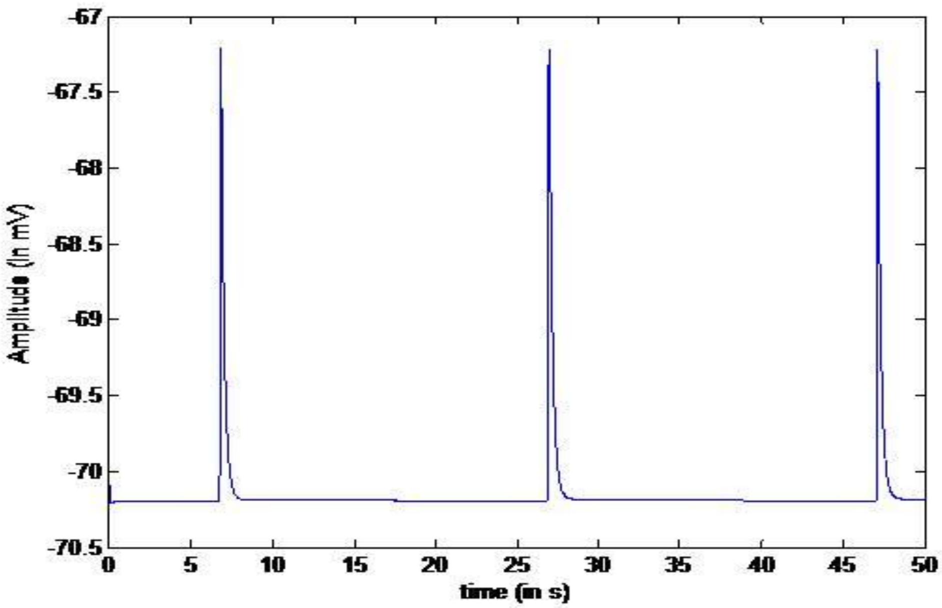


Figure 5: MATLAB: Unitary Potentials (Amplitude 3 mV pk-pk)

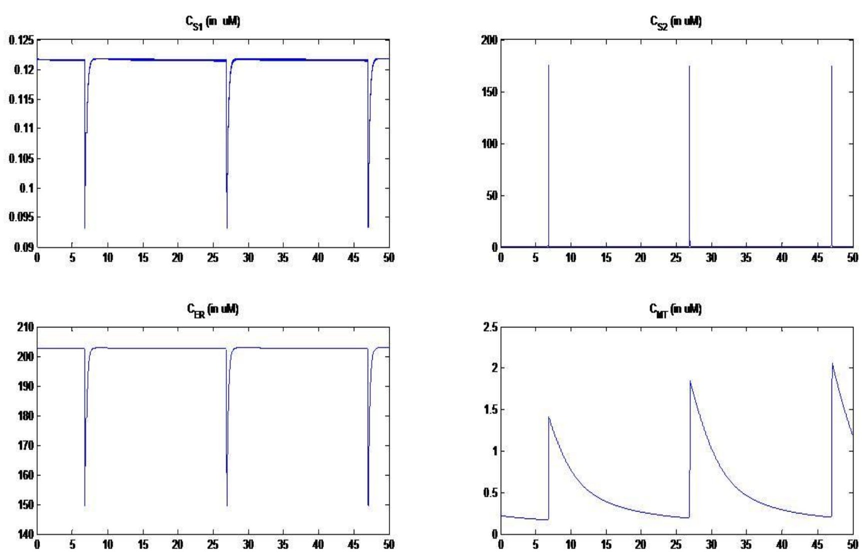


Figure 6: MATLAB Ca^{2+} ion flow in various components of the Pacemaker Unit

Figure 7 shows the waveform simulation using VHDL at a clock of 6.4 kHz. 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.

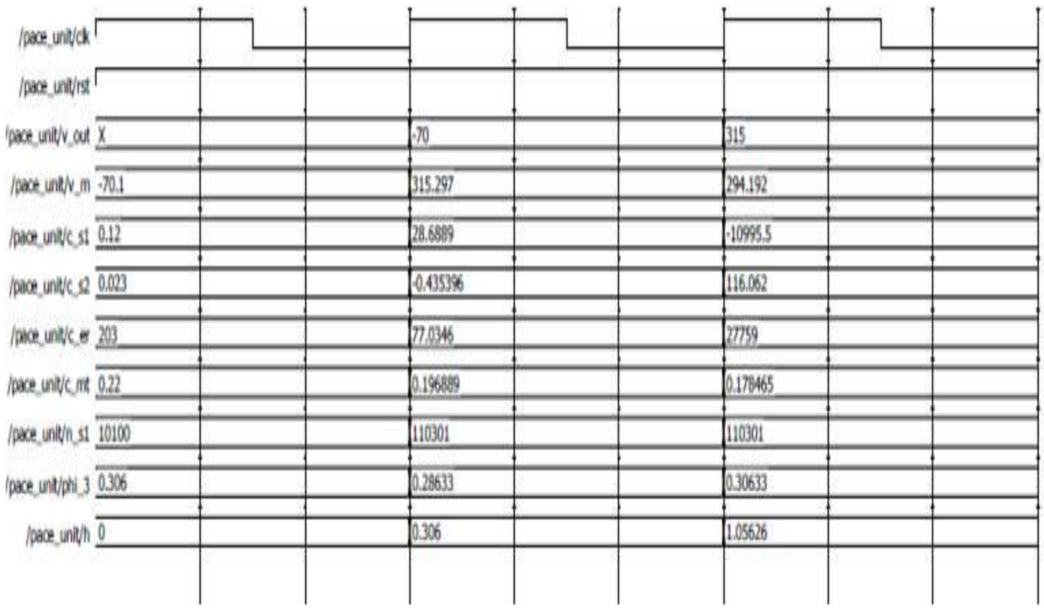


Figure 7: VHDL Simulation Output of the Pacemaker Unit

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

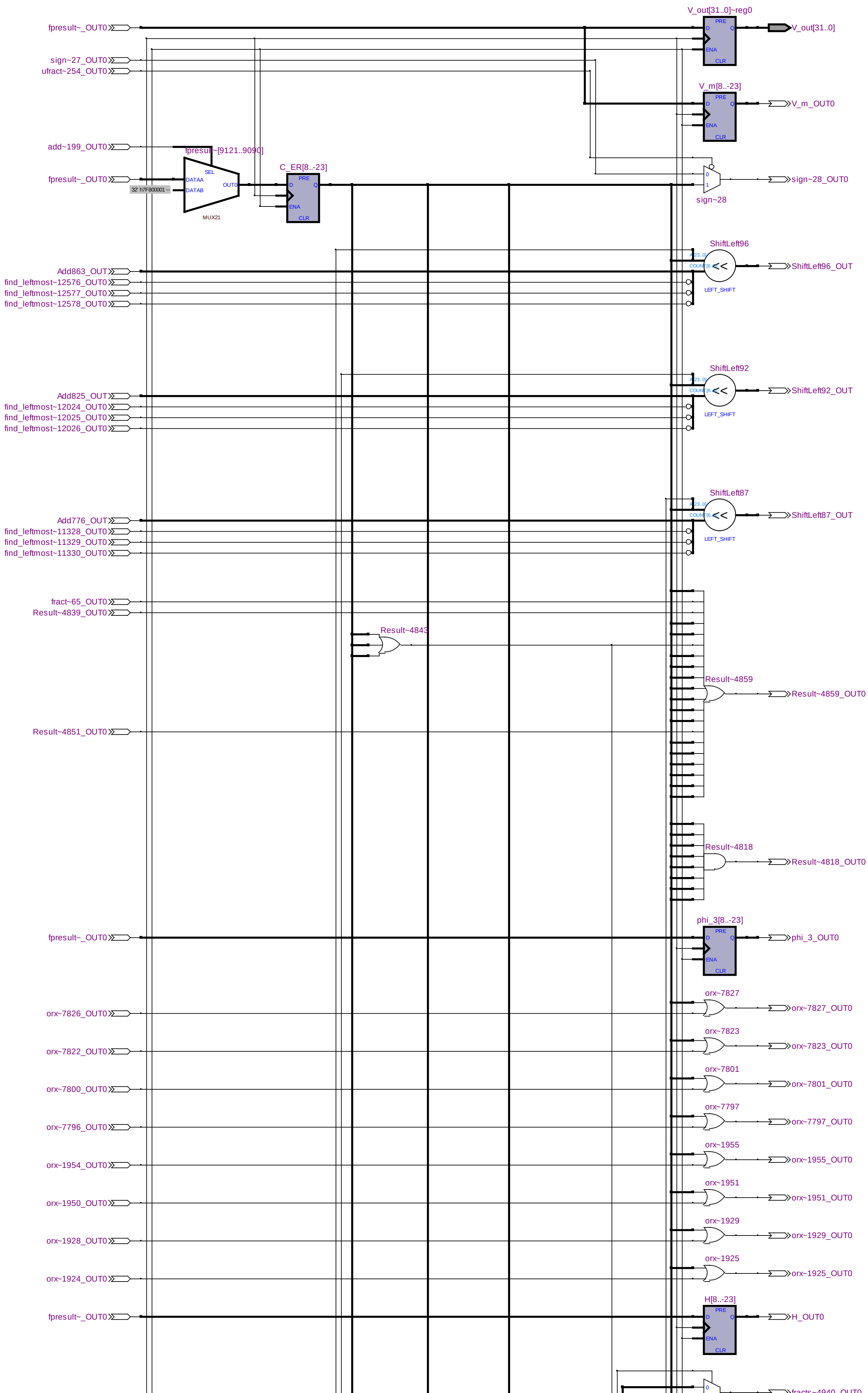
[1] Sanders, K.M., T. Koh, S.D. and Ward., S.M., "Interstitial Cells of Cajal as pacemakers in the gastrointestinal tract", Annu Rev. Physiol. 68:307-343.

[2] Faville, R. , A. Pullan, K. Sanders and N. Smith.2008. A biophysically based mathematical model of unitary potential activity in interstitial cells of Cajal.",Biophysical Journal 95(1):88-104.

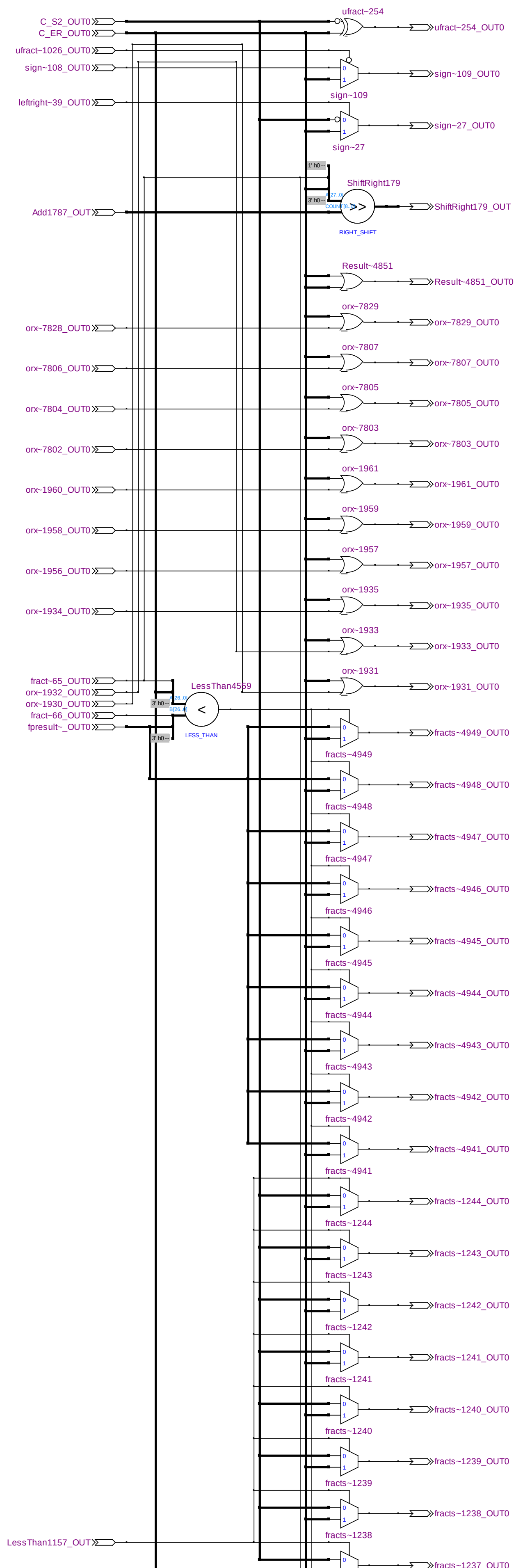
[3] Douglas L. Perry, VHDL- Programming by Example,McGraw-Hill Publishers.

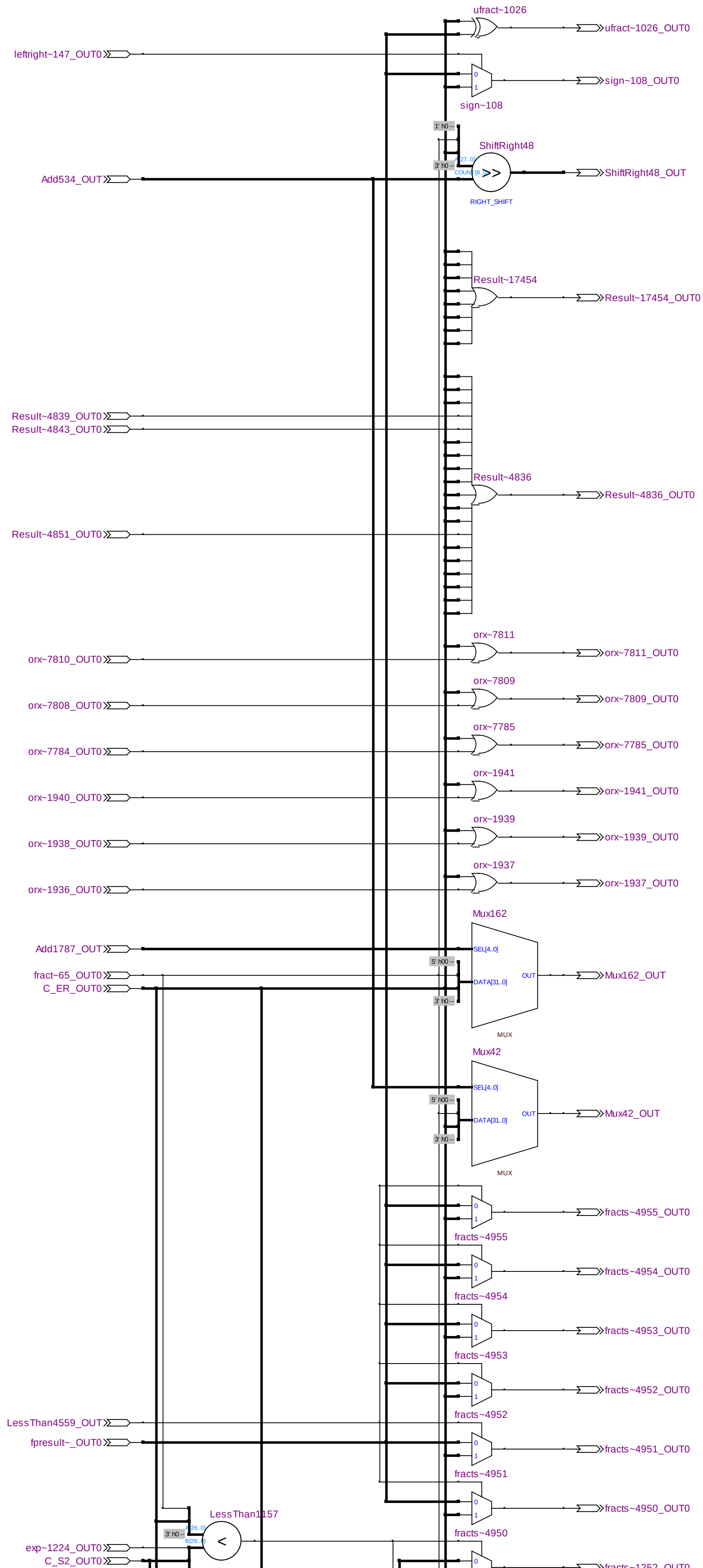
Appendix A

RTL Schematic of the Pacemaker Unit

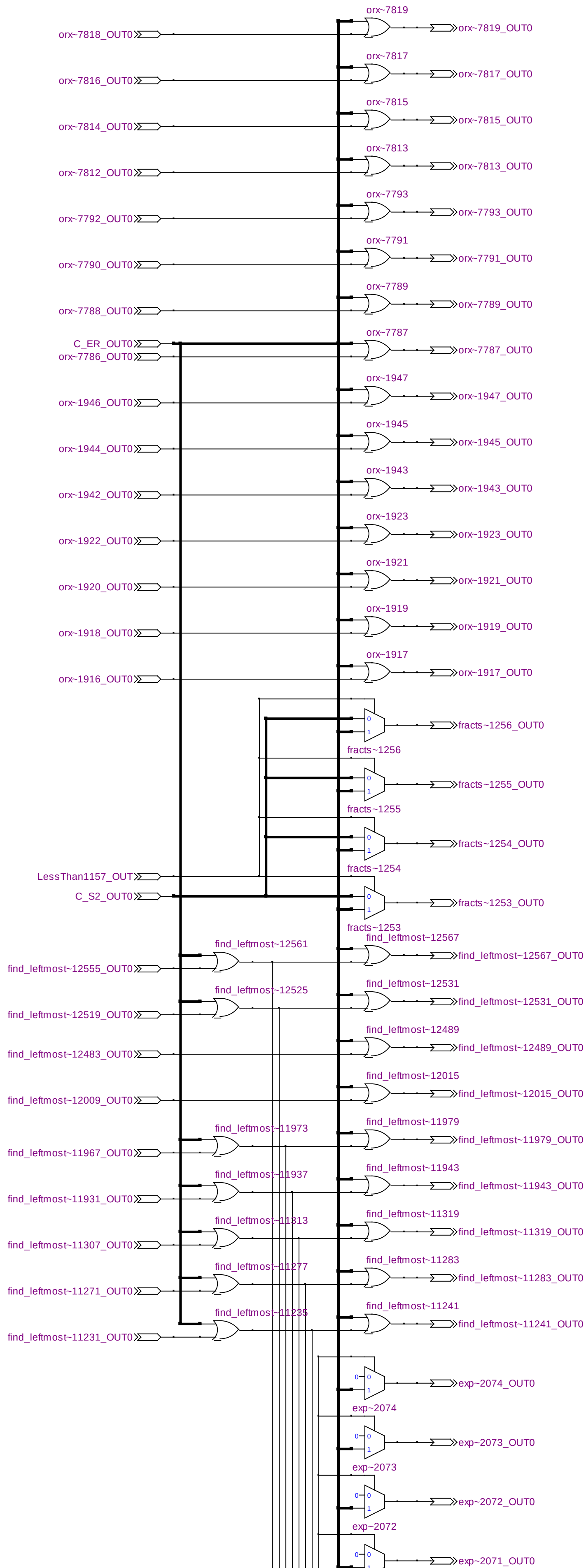


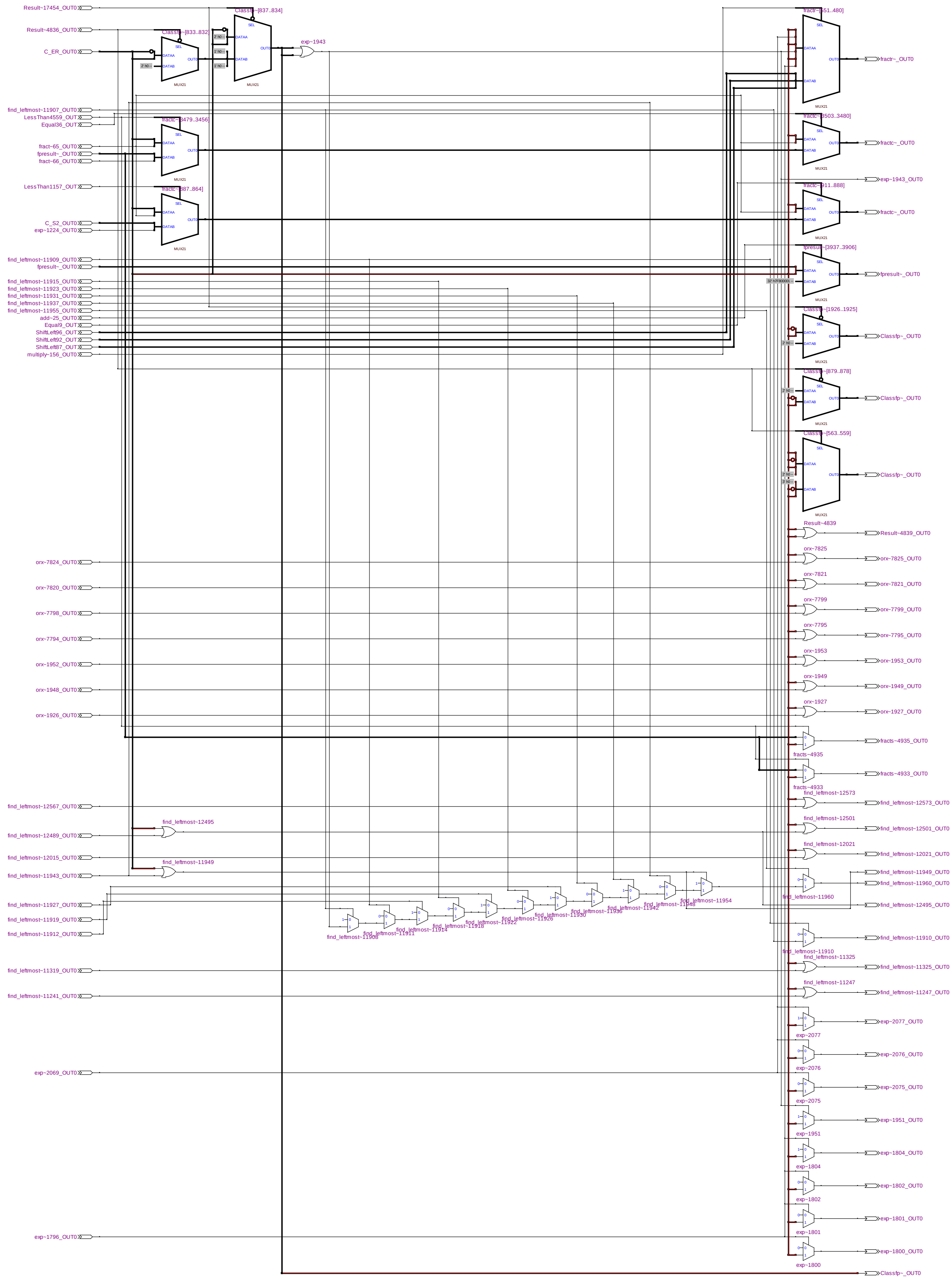
V_out[31..0]~reg0
fpresult~_OUT0
V_out[31..0]
sign~27_OUT0
ufract~254_OUT0
V_m[8..-23]
V_m_OUT0
add~199_OUT0
fpresult~[9121..9090]
C_ER[8..-23]
fpresult~_OUT0
32' h7F800001 --
MUX21
sign~28_OUT0
sign~28
ShiftLeft96
Add863_OUT
find_leftmost~12576_OUT0
find_leftmost~12577_OUT0
find_leftmost~12578_OUT0
ShiftLeft96_OUT
LEFT_SHIFT
ShiftLeft92
Add825_OUT
find_leftmost~12024_OUT0
find_leftmost~12025_OUT0
find_leftmost~12026_OUT0
ShiftLeft92_OUT
ShiftLeft87
Add776_OUT
find_leftmost~11328_OUT0
find_leftmost~11329_OUT0
find_leftmost~11330_OUT0
ShiftLeft87_OUT
fract~65_OUT0
Result~4839_OUT0
Result~4843
Result~4859
Result~4859_OUT0
Result~4851_OUT0
Result~4818
Result~4818_OUT0
phi_3[8..-23]
fpresult~_OUT0
phi_3_OUT0
orx~7827
orx~7826_OUT0
orx~7827_OUT0
orx~7823
orx~7822_OUT0
orx~7823_OUT0
orx~7801
orx~7800_OUT0
orx~7801_OUT0
orx~7797
orx~7796_OUT0
orx~7797_OUT0
orx~1955
orx~1954_OUT0
orx~1955_OUT0
orx~1951
orx~1950_OUT0
orx~1951_OUT0
orx~1929
orx~1928_OUT0
orx~1929_OUT0
orx~1925
orx~1924_OUT0
orx~1925_OUT0
H[8..-23]
fpresult~_OUT0
H_OUT0

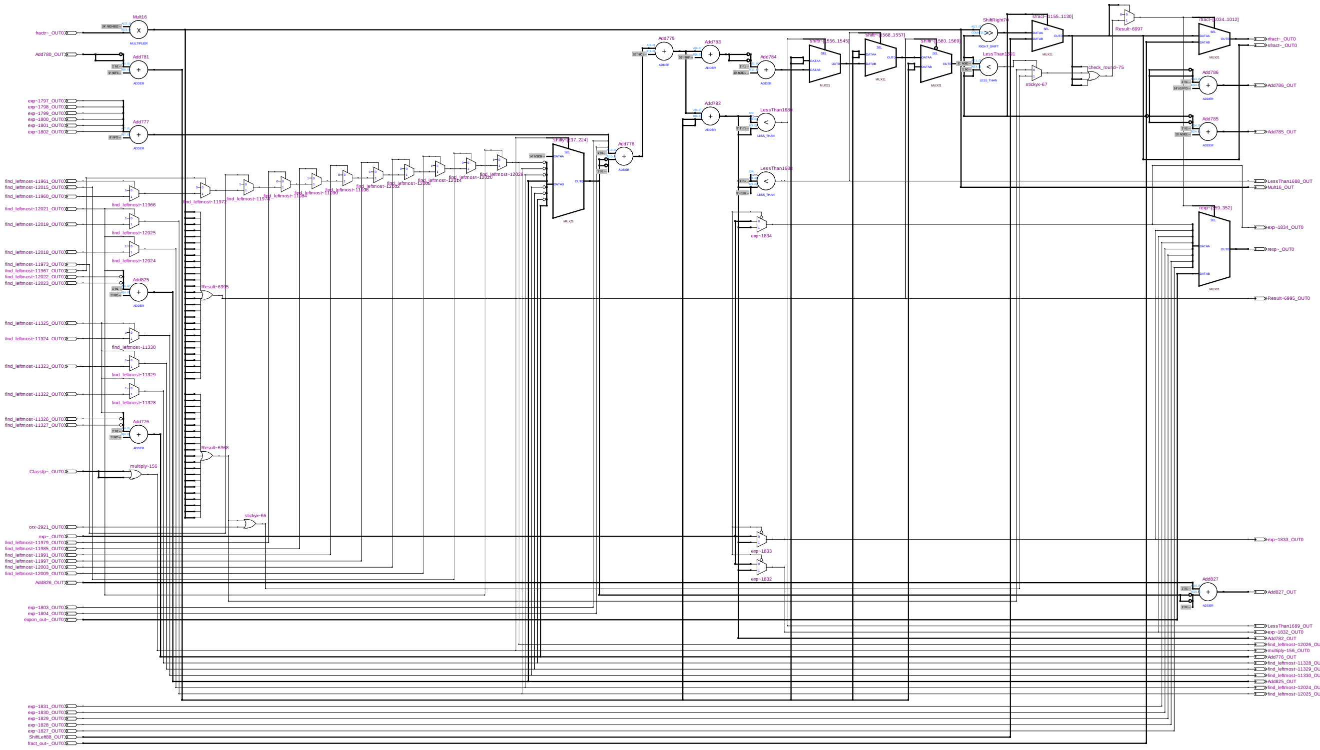

C_S2_OUT0
C_ER_OUT0
ufract~1026_OUT0
sign~108_OUT0
leftright~39_OUT0
Add1787_OUT
orx~7828_OUT0
orx~7806_OUT0
orx~7804_OUT0
orx~7802_OUT0
orx~1960_OUT0
orx~1958_OUT0
orx~1956_OUT0
orx~1934_OUT0
fract~65_OUT0
orx~1932_OUT0
orx~1930_OUT0
fract~66_OUT0
fpresult~_OUT0
LessThan1157_OUT
ufract~254
ufract~254_OUT0
sign~109_OUT0
sign~109
sign~27_OUT0
sign~27
ShiftRight179
ShiftRight179_OUT
RIGHT_SHIFT
Result~4851
Result~4851_OUT0
orx~7829
orx~7829_OUT0
orx~7807
orx~7807_OUT0
orx~7805
orx~7805_OUT0
orx~7803
orx~7803_OUT0
orx~1961
orx~1961_OUT0
orx~1959
orx~1959_OUT0
orx~1957
orx~1957_OUT0
orx~1935
orx~1935_OUT0
orx~1933
orx~1933_OUT0
orx~1931
orx~1931_OUT0
LessThan4559
LESS_THAN
fracts~4949_OUT0
fracts~4949
fracts~4948_OUT0
fracts~4948
fracts~4947_OUT0
fracts~4947
fracts~4946_OUT0
fracts~4946
fracts~4945_OUT0
fracts~4945
fracts~4944_OUT0
fracts~4944
fracts~4943_OUT0
fracts~4943
fracts~4942_OUT0
fracts~4942
fracts~4941_OUT0
fracts~4941
fracts~1244_OUT0
fracts~1244
fracts~1243_OUT0
fracts~1243
fracts~1242_OUT0
fracts~1242
fracts~1241_OUT0
fracts~1241
fracts~1240_OUT0
fracts~1240
fracts~1239_OUT0
fracts~1239
fracts~1238_OUT0
fracts~1238
fracts~1237_OUT0

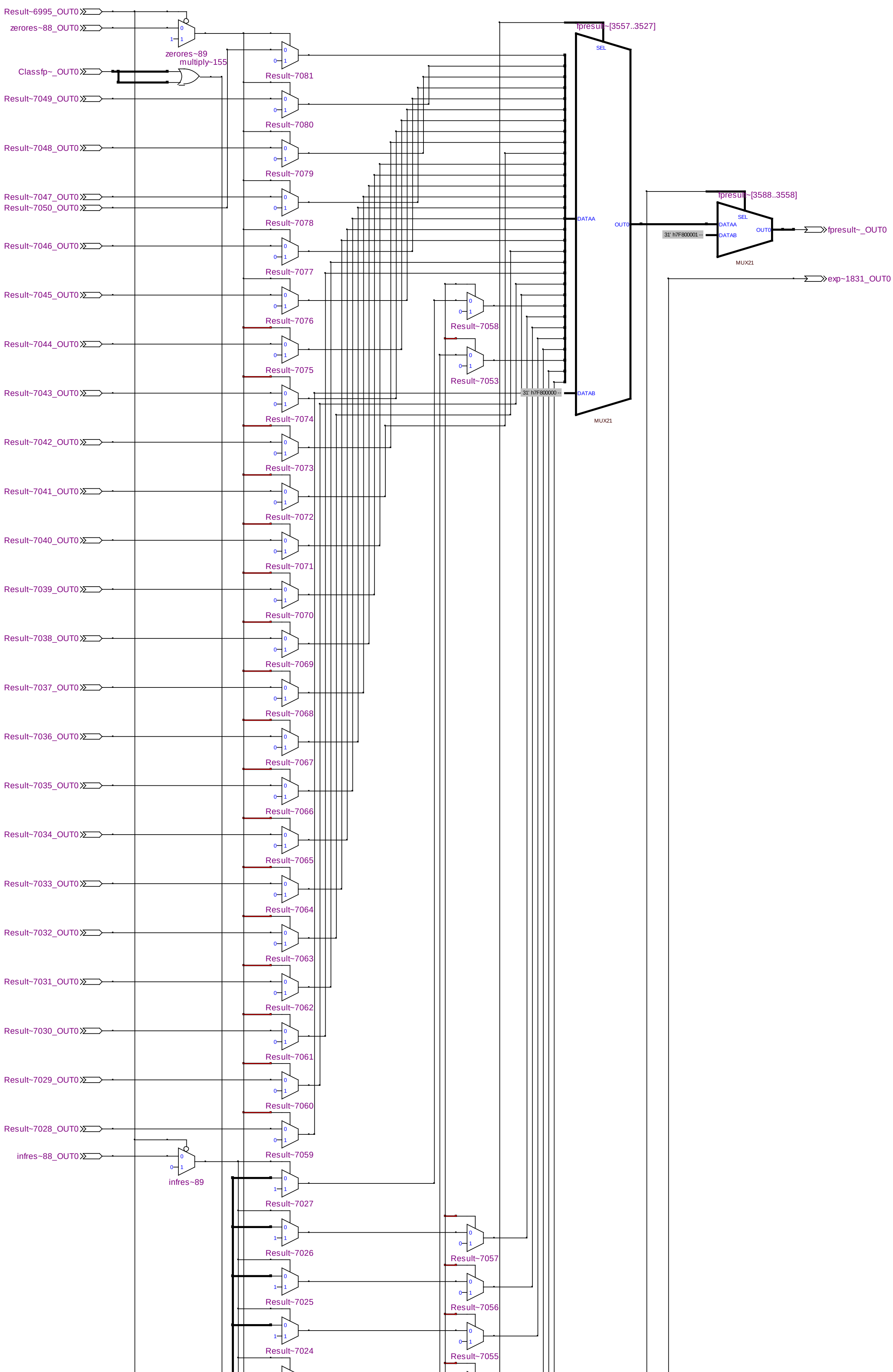


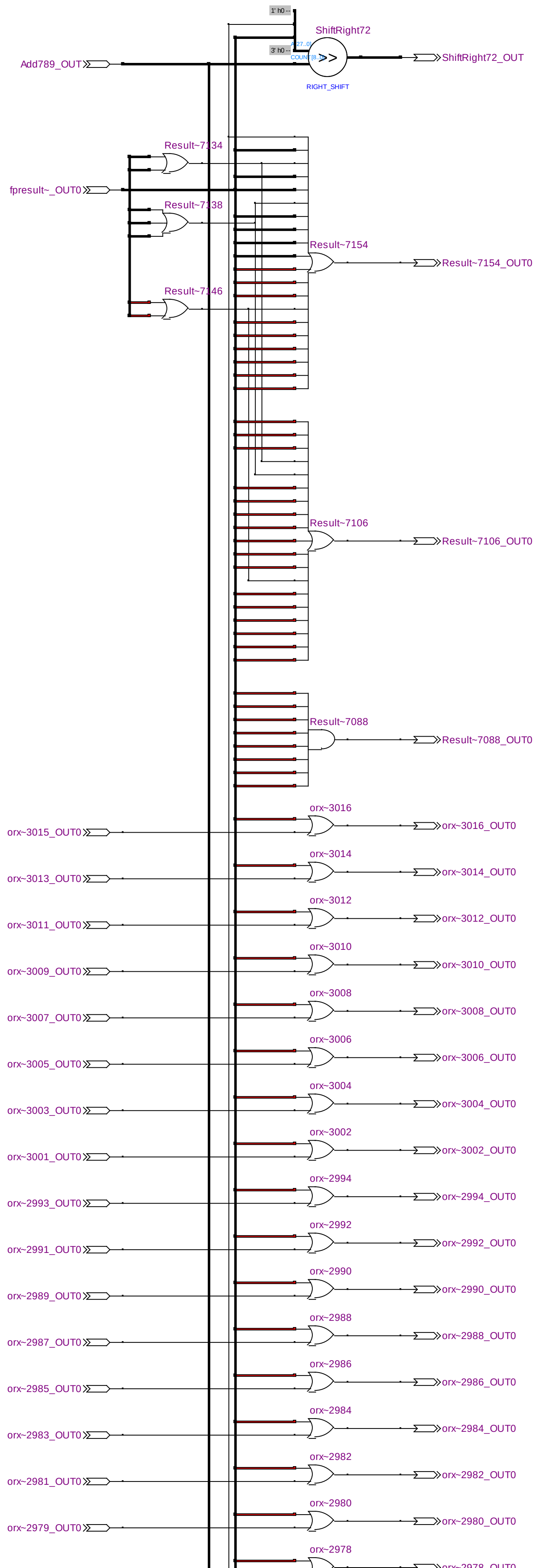
ufract~1026
ufract~1026_OUT0
leftright~147_OUT0
sign~108_OUT0
sign~108
ShiftRight48
Add534_OUT
ShiftRight48_OUT
RIGHT_SHIFT
Result~17454
Result~17454_OUT0
Result~4839_OUT0
Result~4843_OUT0
Result~4836
Result~4836_OUT0
Result~4851_OUT0
orx~7811
orx~7810_OUT0
orx~7811_OUT0
orx~7809
orx~7808_OUT0
orx~7809_OUT0
orx~7785
orx~7784_OUT0
orx~7785_OUT0
orx~1941
orx~1940_OUT0
orx~1941_OUT0
orx~1939
orx~1938_OUT0
orx~1939_OUT0
orx~1937
orx~1936_OUT0
orx~1937_OUT0
Mux162
Add1787_OUT
fract~65_OUT0
C_ER_OUT0
Mux162_OUT
MUX
Mux42
Mux42_OUT
MUX
fracts~4955_OUT0
fracts~4955
fracts~4954_OUT0
fracts~4954
fracts~4953_OUT0
fracts~4953
fracts~4952_OUT0
fracts~4952
LessThan4559_OUT
fpresult~_OUT0
fracts~4951_OUT0
fracts~4951
fracts~4950_OUT0
LessThan1157
fracts~4950
exp~1224_OUT0
C_S2_OUT0


orx~7819
orx~7818_OUT0
orx~7819_OUT0
orx~7817
orx~7816_OUT0
orx~7817_OUT0
orx~7815
orx~7814_OUT0
orx~7815_OUT0
orx~7813
orx~7812_OUT0
orx~7813_OUT0
orx~7793
orx~7792_OUT0
orx~7793_OUT0
orx~7791
orx~7790_OUT0
orx~7791_OUT0
orx~7789
orx~7788_OUT0
orx~7789_OUT0
orx~7787
C_ER_OUT0
orx~7786_OUT0
orx~7787_OUT0
orx~1947
orx~1946_OUT0
orx~1947_OUT0
orx~1945
orx~1944_OUT0
orx~1945_OUT0
orx~1943
orx~1942_OUT0
orx~1943_OUT0
orx~1923
orx~1922_OUT0
orx~1923_OUT0
orx~1921
orx~1920_OUT0
orx~1921_OUT0
orx~1919
orx~1918_OUT0
orx~1919_OUT0
orx~1917
orx~1916_OUT0
orx~1917_OUT0
fracts~1256_OUT0
fracts~1256
fracts~1255_OUT0
fracts~1255
fracts~1254_OUT0
LessThan1157_OUT
fracts~1254
C_S2_OUT0
fracts~1253_OUT0
fracts~1253
find_leftmost~12561
find_leftmost~12567
find_leftmost~12555_OUT0
find_leftmost~12567_OUT0
find_leftmost~12525
find_leftmost~12531
find_leftmost~12519_OUT0
find_leftmost~12531_OUT0
find_leftmost~12489
find_leftmost~12483_OUT0
find_leftmost~12489_OUT0
find_leftmost~12015
find_leftmost~12009_OUT0
find_leftmost~12015_OUT0
find_leftmost~11973
find_leftmost~11979
find_leftmost~11967_OUT0
find_leftmost~11979_OUT0
find_leftmost~11937
find_leftmost~11943
find_leftmost~11931_OUT0
find_leftmost~11943_OUT0
find_leftmost~11313
find_leftmost~11319
find_leftmost~11307_OUT0
find_leftmost~11319_OUT0
find_leftmost~11277
find_leftmost~11283
find_leftmost~11271_OUT0
find_leftmost~11283_OUT0
find_leftmost~11235
find_leftmost~11241
find_leftmost~11231_OUT0
find_leftmost~11241_OUT0
exp~2074_OUT0
exp~2074
exp~2073_OUT0
exp~2073
exp~2072_OUT0
exp~2072
exp~2071_OUT0

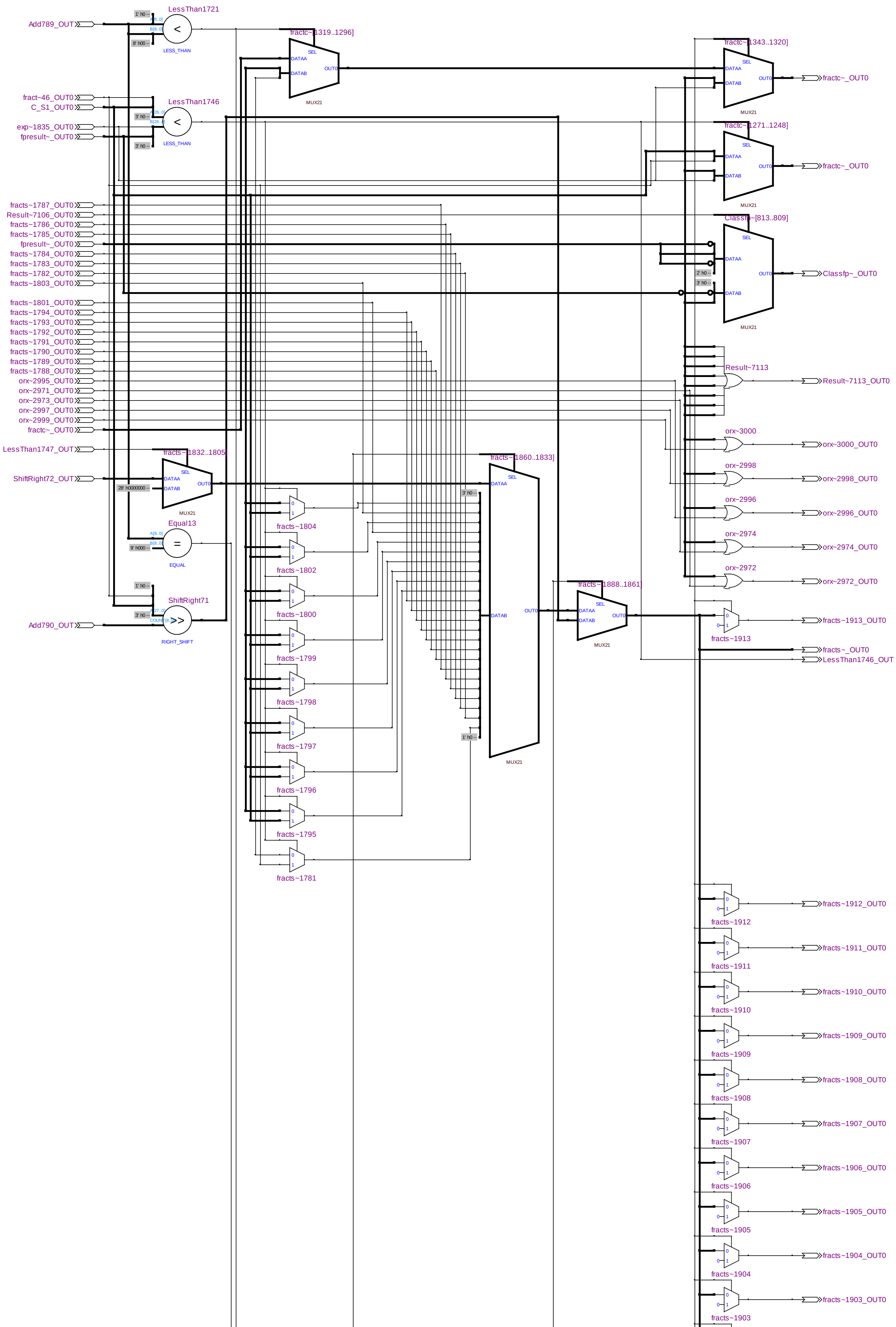

Result~17454_OUT0
Result~4836_OUT0
C_ER_OUT0
find_leftmost~11907_OUT0
LessThan4559_OUT0
Equal36_OUT0
fract~65_OUT0
fpresult~_OUT0
fract~66_OUT0
LessThan1157_OUT0
C_S2_OUT0
exp~1224_OUT0
find_leftmost~11909_OUT0
fpresult~_OUT0
find_leftmost~11915_OUT0
find_leftmost~11923_OUT0
find_leftmost~11931_OUT0
find_leftmost~11937_OUT0
find_leftmost~11955_OUT0
add~25_OUT0
Equal9_OUT0
ShiftLeft96_OUT0
ShiftLeft92_OUT0
ShiftLeft87_OUT0
multiply~156_OUT0
orx~7824_OUT0
orx~7820_OUT0
orx~7798_OUT0
orx~7794_OUT0
orx~1952_OUT0
orx~1948_OUT0
orx~1926_OUT0
find_leftmost~12567_OUT0
find_leftmost~12489_OUT0
find_leftmost~12015_OUT0
find_leftmost~11943_OUT0
find_leftmost~11927_OUT0
find_leftmost~11919_OUT0
find_leftmost~11912_OUT0
find_leftmost~11319_OUT0
find_leftmost~11241_OUT0
exp~2069_OUT0
exp~1796_OUT0
Classfp~[833..832]
Classfp~[837..834]
exp~1943
fractc~[3479..3456]
fractc~[887..864]
MUX21
fractr~[551..480]
fractc~[3503..3480]
fractc~[911..888]
fpresult~[3937..3906]
Classfp~[1926..1925]
Classfp~[879..878]
Classfp~[563..559]
Result~4839
orx~7825
orx~7821
orx~7799
orx~7795
orx~1953
orx~1949
orx~1927
fracts~4935
fracts~4933
find_leftmost~12573
find_leftmost~12501
find_leftmost~12021
find_leftmost~12495
find_leftmost~11949
find_leftmost~11960
find_leftmost~11910
find_leftmost~11325
find_leftmost~11247
exp~2077
exp~2076
exp~2075
exp~1951
exp~1804
exp~1802
exp~1801
exp~1800
fractr~_OUT0
fractc~_OUT0
exp~1943_OUT0
fractc~_OUT0
fpresult~_OUT0
Classfp~_OUT0
Classfp~_OUT0
Classfp~_OUT0
Result~4839_OUT0
orx~7825_OUT0
orx~7821_OUT0
orx~7799_OUT0
orx~7795_OUT0
orx~1953_OUT0
orx~1949_OUT0
orx~1927_OUT0
fracts~4935_OUT0
fracts~4933_OUT0
find_leftmost~12573_OUT0
find_leftmost~12501_OUT0
find_leftmost~12021_OUT0
find_leftmost~11949_OUT0
find_leftmost~11960_OUT0
find_leftmost~12495_OUT0
find_leftmost~11910_OUT0
find_leftmost~11325_OUT0
find_leftmost~11247_OUT0
exp~2077_OUT0
exp~2076_OUT0
exp~2075_OUT0
exp~1951_OUT0
exp~1804_OUT0
exp~1802_OUT0
exp~1801_OUT0
exp~1800_OUT0
Classfp~_OUT0



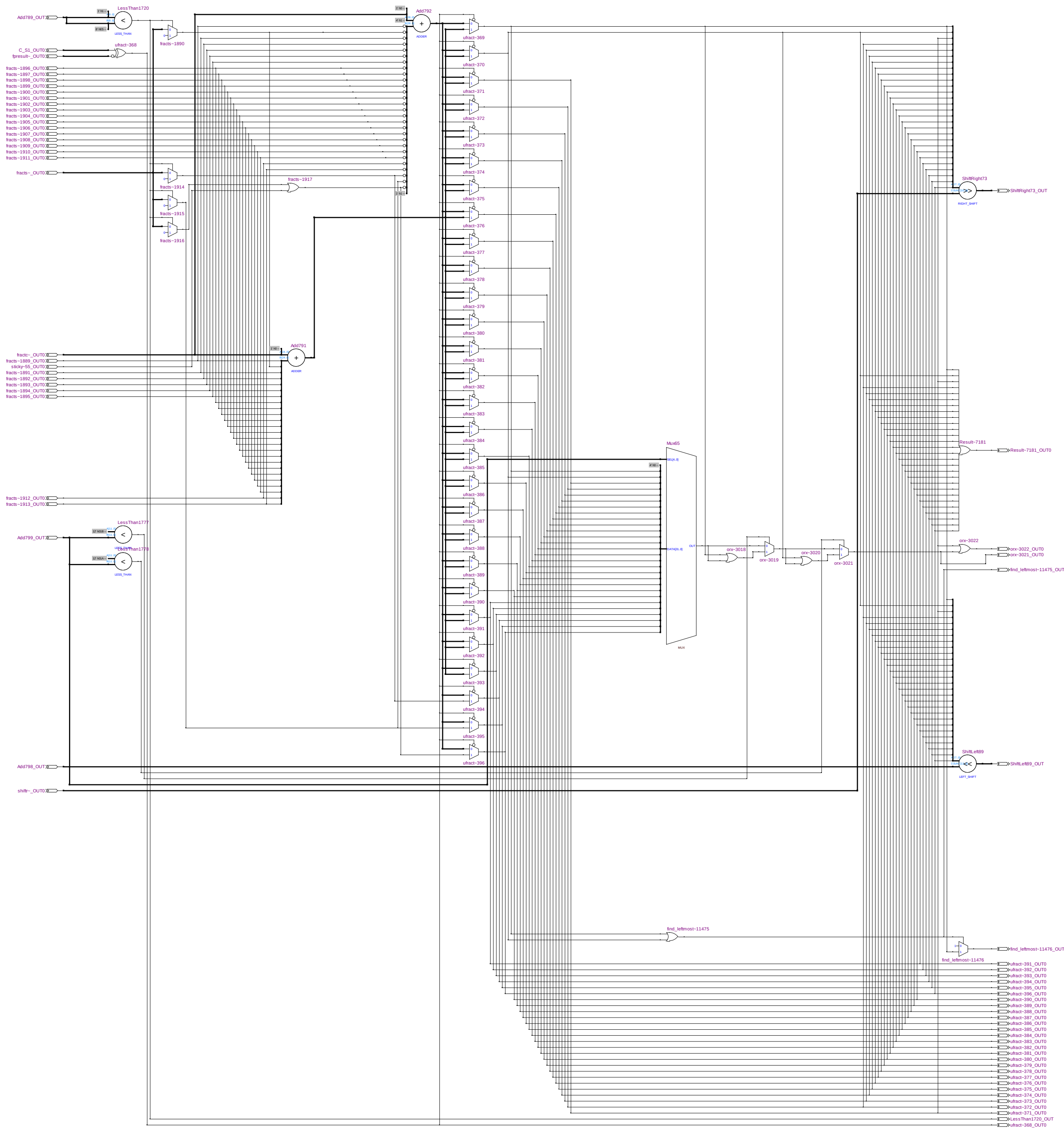

Result~6995_OUT0
zerores~88_OUT0
zerores~89
multiply~155
Classfp~_OUT0
Result~7049_OUT0
Result~7048_OUT0
Result~7047_OUT0
Result~7050_OUT0
Result~7046_OUT0
Result~7045_OUT0
Result~7044_OUT0
Result~7043_OUT0
Result~7042_OUT0
Result~7041_OUT0
Result~7040_OUT0
Result~7039_OUT0
Result~7038_OUT0
Result~7037_OUT0
Result~7036_OUT0
Result~7035_OUT0
Result~7034_OUT0
Result~7033_OUT0
Result~7032_OUT0
Result~7031_OUT0
Result~7030_OUT0
Result~7029_OUT0
Result~7028_OUT0
infres~88_OUT0
infres~89
Result~7081
Result~7080
Result~7079
Result~7078
Result~7077
Result~7076
Result~7075
Result~7074
Result~7073
Result~7072
Result~7071
Result~7070
Result~7069
Result~7068
Result~7067
Result~7066
Result~7065
Result~7064
Result~7063
Result~7062
Result~7061
Result~7060
Result~7059
Result~7027
Result~7026
Result~7025
Result~7024
Result~7058
Result~7053
Result~7057
Result~7056
Result~7055
fpresult~[3557..3527]
SEL
DATAA
OUT0
DATAB
31' h7F800000 --
MUX21
fpresult~[3588..3558]
31' h7F800001 --
fpresult~_OUT0
exp~1831_OUT0



ShiftRight72
Add789_OUT
ShiftRight72_OUT
RIGHT_SHIFT
Result~7134
fpresult~_OUT0
Result~7138
Result~7154
Result~7154_OUT0
Result~7146
Result~7106
Result~7106_OUT0
Result~7088
Result~7088_OUT0
orx~3016
orx~3015_OUT0
orx~3016_OUT0
orx~3014
orx~3013_OUT0
orx~3014_OUT0
orx~3012
orx~3011_OUT0
orx~3012_OUT0
orx~3010
orx~3009_OUT0
orx~3010_OUT0
orx~3008
orx~3007_OUT0
orx~3008_OUT0
orx~3006
orx~3005_OUT0
orx~3006_OUT0
orx~3004
orx~3003_OUT0
orx~3004_OUT0
orx~3002
orx~3001_OUT0
orx~3002_OUT0
orx~2994
orx~2993_OUT0
orx~2994_OUT0
orx~2992
orx~2991_OUT0
orx~2992_OUT0
orx~2990
orx~2989_OUT0
orx~2990_OUT0
orx~2988
orx~2987_OUT0
orx~2988_OUT0
orx~2986
orx~2985_OUT0
orx~2986_OUT0
orx~2984
orx~2983_OUT0
orx~2984_OUT0
orx~2982
orx~2981_OUT0
orx~2982_OUT0
orx~2980
orx~2979_OUT0
orx~2980_OUT0
orx~2978



LessThan1721
LESS_THAN
fractc~[1319..1296]
MUX21
fractc~[1343..1320]
MUX21
fractc~_OUT0
LessThan1746
LESS_THAN
fractc~[1271..1248]
MUX21
fractc~_OUT0
Classfp~[813..809]
MUX21
Classfp~_OUT0
Add789_OUT
fract~46_OUT0
C_S1_OUT0
exp~1835_OUT0
fpresult~_OUT0
fracts~1787_OUT0
Result~7106_OUT0
fracts~1786_OUT0
fracts~1785_OUT0
fpresult~_OUT0
fracts~1784_OUT0
fracts~1783_OUT0
fracts~1782_OUT0
fracts~1803_OUT0
fracts~1801_OUT0
fracts~1794_OUT0
fracts~1793_OUT0
fracts~1792_OUT0
fracts~1791_OUT0
fracts~1790_OUT0
fracts~1789_OUT0
fracts~1788_OUT0
orx~2995_OUT0
orx~2971_OUT0
orx~2973_OUT0
orx~2997_OUT0
orx~2999_OUT0
fractc~_OUT0
LessThan1747_OUT
ShiftRight72_OUT
Add790_OUT
Result~7113
Result~7113_OUT0
orx~3000
orx~3000_OUT0
orx~2998
orx~2998_OUT0
orx~2996
orx~2996_OUT0
orx~2974
orx~2974_OUT0
orx~2972
orx~2972_OUT0
fracts~[1832..1805]
MUX21
Equal13
EQUAL
ShiftRight71
RIGHT_SHIFT
fracts~[1860..1833]
MUX21
fracts~[1888..1861]
MUX21
fracts~1804
fracts~1802
fracts~1800
fracts~1799
fracts~1798
fracts~1797
fracts~1796
fracts~1795
fracts~1781
fracts~1913
fracts~1913_OUT0
fracts~_OUT0
LessThan1746_OUT
fracts~1912
fracts~1912_OUT0
fracts~1911
fracts~1911_OUT0
fracts~1910
fracts~1910_OUT0
fracts~1909
fracts~1909_OUT0
fracts~1908
fracts~1908_OUT0
fracts~1907
fracts~1907_OUT0
fracts~1906
fracts~1906_OUT0
fracts~1905
fracts~1905_OUT0
fracts~1904
fracts~1904_OUT0
fracts~1903
fracts~1903_OUT0



210

211

212

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;
constant 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;
constant 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 := 10100.0;
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);--IP3R Slow Rate Variable
                --IP3R Constants
                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));
                I_Na <= g_PM*(C_S1**h_Na/(K_Na**h_Na+C_S1**h_Na));
                g_NSCC_Ca <=
g_capNSCC_Ca*(K_NSCC**h_NSCC/(K_NSCC**h_NSCC+C_S1**h_NSCC));
          g_NSCC_Na <=
g_capNSCC_Na*(K_NSCC**h_NSCC/(K_NSCC**h_NSCC+C_S1**h_NSCC));
          I_NSCC_Ca <= g_NSCC_Ca*(V_m-E_NSCC);
                I_NSCC_Na <= g_NSCC_Na*(V_m-E_NSCC);
                E_Ca <= 0.5*V_t*log(C_O/C_S1);
                I_Ca <= g_Ca*(V_m-E_Ca);
                l_MTS1 <= y_MT/y_S1;
                l_ERS1 <= y_ER/y_S1;
                l_ERS2 <= y_ER/y_S2;
                l_S1S2 <= y_S1/y_S2;
```

```
				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_NCX)-
((d_S/Z_Ca)*I_iCa+l_ERS1*J_SERCA);
				C_S2 <= C_S2 + (l_ERS2*J_IPR-(l_S1S2*J_S1S2+l_MTS2*J_MCU));
				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;
```