Research Article

A pulse sensor interface design for FPGA based multisensor health monitoring platform

Abstract

The FPGA-based platform is critical for producing an inexpensive early validation platform design. In past years, sensor nodes based on the FPGA platform have been proposed to be IoT low-end devices. In this study, we present the FPGA based IoT low-end reconfigurable pulse sensor interface design that can be integrated with a multi-sensor healthcare platform to monitor a human pulse vital sign and be able to distinguish between user normal, Bradycardia, or Tachycardia heart rate. The pulse sensor interface is implemented by VHDL programming and FPGA technology. The designed pulse sensor peripheral interface is reliable and reconfigurable. It can collect vital body signs with the accuracy of a 15nanoseconds period. The peripheral in FPGAs embedded system has been tested by placing the biosensor on the user's fingertips. The BPM can be updated every 15seconds.

Keywords: FPGA platform, IoT low-end devices, pulse sensor, VHDL programming, peripheral interface

Volume 5 Issue 1 - 2019

Yu Wang, Sunghoon Jang
Department of Computer Engineering Technology, The New York City College of Technology, USA

Correspondence: Yu Wang, Department of Computer Engineering Technology, the New York City College of Technology of the City University of New York, Brooklyn, NY, 11201, USA, Email YWang@citytech.cuny.edu

Received: January 31, 2017 | **Published:** February 18, 2019

Introduction

In the 21st century, IoT (Internet of Things) technology is making daily life more convenient via heterogeneous smart devices through seamless connectivity. Tens of billions of new IoT low-end devices are coupled with sensors and are connected to the Internet with an open source operating system, with limited memory, limited computational power, and limited power supplies.[1–4] For IoT devices and platforms of CPUs (central processing units), embedded GPUs (graphics processing units), and FPGAs (field-programmable gate arrays), the cost, power, performance, and volume constraints have to be tradeoff.[5,6] The architecture of the FPGA makes it possible to implement any combinational and sequential circuits, which can run independently at different frequencies than the microcontroller. It can be reconfigured into a simple logic function, a master controller, and a soft processor. CAD tools or Hardware Description Languages (HDL) such as VHDL and Verilog HDL are used to program the FPGA. The survey in[7] shows low-power optimized FPGAs can enhance the computation of several types of algorithms in terms of speed and power consumption in comparison to microcontrollers of commercial sensor nodes. In past years, sensor nodes based on the FPGA and CPLD platform have been proposed to be IoT low-end devices.[8,9] The FPGA-based platform is critical for producing an inexpensive early validation of the platform design. An ultra-low-power and reliable FPGA is a promising solution for IoT applications. An FPGA-based edge device for IoT was proposed for dedicated hardware.[8] The proposed stack can be implemented on all IoT devices avoiding the battle for the wireless standard. The reconfigurable sensor interface for wireless sensor networks in an IoT environment was proposed in.[9] The core controller adopts a complex programmable logic device (CPLD) that can read data in parallel and in real time with high speed on multiple different sensor data. Such a trend has prospects of becoming integrated into biosensor and FPGAs platform in health care systems.[10–12] The biosensors integrated with the FPGA platform can monitor and measure the human body temperature, heart rate, and respiratory rate with an accuracy up to 96%. In,[11] fully digital time-domain temperature sensors driven by five pulse-generators are designed and implemented in the FPGA platform. An energy efficient hardware model was implemented on the FPGA,[12] This model is used to compress and reconstruct body vital signs data, for example, EEG and ECG, based on discrete wavelet transform (DWT). These research projects contribute to the advancement of the engineering field because it proves that the IoT–based smart healthcare network can be built efficiently with FPGA devices and will enhance the health care of the user.

We propose to design the FPGA based IoT low-end interface for the health monitoring environment. In the future, this FPGA based controller will connect to our platform[13,14] including the microcontroller, Raspberry Pi, Bluetooth, Wi-Fi, and Firebase to monitor human heartbeats and relay user's normal, Bradycardia, or Tachycardia heart rate back to caregivers and other medical devices that can be converted or deployed as IoT technology. The designed FPGA based pulse sensor interface can be driven by digital components such as AND, OR gates, Multiplexer, Counters, Frequency Divider, etc. Each component is implemented with VHDL programming. With system clock frequency of 450MHz on Nexys 4 FPGA board,[15] we can derive a 100MHz clock, which is further used to read body vital signs with an accuracy of a 15ns period.

The principle of operation of the pulse sensor

The heart rate is a vital body sign. Heart rate varies depending on a person's body. Physical exercise, sleep, anxiety, stress, and illness affect the heart rate. An average heart rate is from 60 to 100 beats per minute (BPM). Arrhythmia can cause heat to beat slower than 60BPM (Bradycardia) or faster than 100BPM (Tachycardia). Light is absorbed well in blood and weakly absorbed in tissue. When a heart beats, the blood flow makes more absorption of the wavelength of green light than other wavelengths. The pulse sensor in Figure 1 can obtain heart beating data from the users by placing the sensor on their fingertips. The hardware schematic of the pulse sensor (Figure 2) includes ambient light sensors APDS-9008, green super bright LED AM2520ZGC09, and an operational amplifier MCP 6001.[16] The optical sensors APDS-9008[17] has excellent responsivity to the wavelength range of 500nm ~ 600nm with the peak sensitivity of green light 565nm. The saturation output voltage of APDS-9008 can reach up to 1.6V for a load of R=12KΩ when power supply is V_{DD}=1.8V. The green LED light AM2520ZGC09[18] has the dominant wavelength of 525nm that match the responsibility of the light sensors APDS-9008. The MCP6001 is a single general purpose operational amplifier offering rail-to-rail input and output voltage between 1.8V and 6V.[19] When the pulse sensor is amped in close contact with the fingertip or earlobe (or other contact points), the AM2520ZGC09 LED emits green light to the fingertip or earlobe and the reflected green light will be sensed by the sensor APDS-9008. When blood pumps through tissues, the converted voltage output will change due to green light absorption. The voltage variation then passes to the equipped filter and amplifier MCP6001. The amplified signal from op amp MCP6001 can achieve the voltage output between 1.8 V and 6V. This output voltage level is sufficient for a digital "high" logic state. The logical state allows us to connect the pulse sensor shown in Figure 1 directly to any reconfigurable digital line of the FPGA. The data being measured are BPM, which is computed and programmed by VHDL programming.

Figure 1 Pulse sensor.

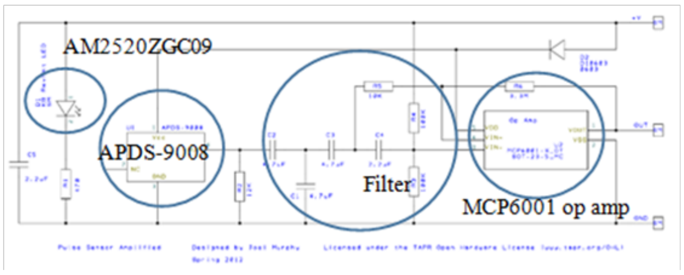

Figure 2 The schematic of the pulse sensor circuit cited from.[16]

FPGA based peripheral interface design

The pulse sensor (Figure 1) is a designed plug-and-play heart-rate analog sensor interfacing with the Arduino project.[16] Since Nexys 4 FPGA board has two voltage levels of 1.8V and 3.3V, the Multisim circuit is created to study if the output signal from the pulse sensor can be directly applied to a digital line of FPGA without using the analog to digital converter. The function generator in the experiment (Figure 3) generates a sinewave (shown at the oscilloscope XSC1) with the frequency f =80Hz, duty cycle d =50%, and amplitude V_{peak}=400mV. The generated sine wave is added to the load of APDS-9008 R=12KΩ under the power supply V_{DD}=1.8V. The output voltage from op-amp MCP6001 shows the rectified pulse signal in the oscilloscope XSC2, which is acting as digital logic "high" and logic "low" with the same period of the input sinewave signal. In Figure 4, a saw tooth waveform signal is generated by the function generator. The signal has a peak amplitude of V_{peak}=1.2V and frequency f =80Hz, which is added to the load resistor R=12KΩ with the power supply V_{DD}=3.3V. Similarly, the output waveform is equivalent to a digital logic "high" and "low" with the same period of the analog saw tooth input signal. Hence the voltage output from the op amp MCP6001 can be directly applied to configured FPGA I/O node. When the sensor senses a heartbeat, the configured FPGA I/O node is reading a signal as a logical high.

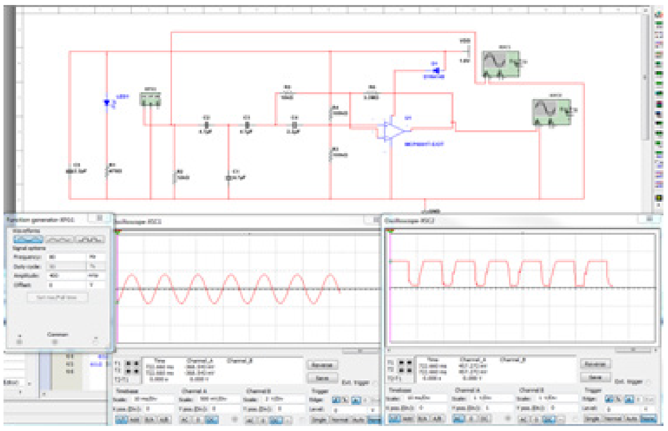
Figure 3 Multisim simulation with sinewave input V_{peak}=400mV and f =80Hz under V_{DD}=1.8V

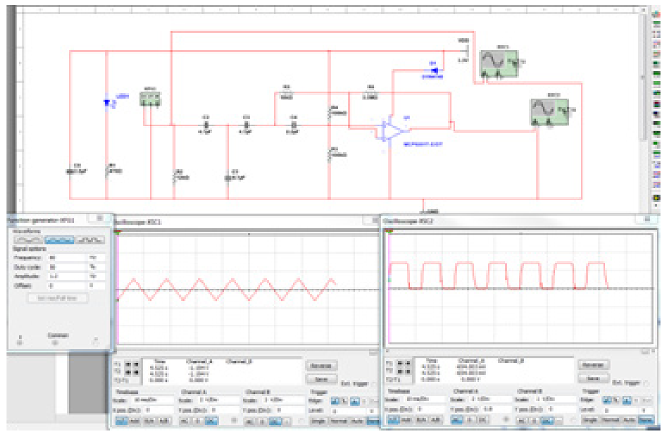
Figure 4 Multisim simulation with saw tooth waveform input V_{peak}=1.2V and f=80Hz under V_{DD}=3.3V

We designed algorithm A (Figure 5) and algorithm B (Figure 6) and programmed the FPGA based interface accordingly. When the received signal from an interface is logical high, which represents a heartbeat. The information of the heartbeat can be collected whenever the signal is high. The period to record the number of high signals can be derived from the system clock of FPGA board through a frequency divider. To display the heart rate BPM on two 7-segment LEDs at a real-time and avoid to wait a whole minute in the event of emergency, algorithm B records the number of logical high signals returned by the pulse sensor within M=15seconds, and multiplies that amount by 4, which gives us the heart rate per minute of the user. The BPM is updated every 15seconds. In the hardware programming code of VHDL, this is accomplished by algorithm A to create a 15seconds counter derived from a 100MHz clock. The internal signal variables are created inside of the architecture part for the interface implementation: COUNTER, PRESCALER, Pulse_COUNT15, and Sensor COUNTER. The COUNTER records the time in seconds from 0 to 15 and will reset to 0 every 15seconds. PRESCALER will records the number of clock ticks to generate 1second timer, the

Pulse_COUNT15 will collect the number of heartbeats during every 15seconds. When a heartbeat is sensed, the Pulse LED will be a logical high and Pulse_COUNT15 is increased by 1. SensorCOUNTER indicates heartbeats per minute where Sensor COUNTER=4 x Pulse_COUNT15. At the circuit level, it is more efficient to have a logical shift and addition than the multiplication. Algorithm B performs a logical left shift of 2bits, which is equivalent to multiplication by 4. In this way, we can eliminate some sources of switching activity to save power consumption.

```
Algorithm A: M seconds Generator
------------------------------------------------------------
Objective: Generate a M =15 seconds timer from given clock
Clock= 100MHz
Convert decimal value Clock (100,000,000) to N=27 bits binary
value BinaryVec= 101111101011110000100000000
Set internal signal PRESCALER with vector dimension N=27 bits
Set internal signal COUNTER to record 4 bits binary number
(0000 ~ 1111)
   for every tick (10ns) of the Clock
      if PRESCALER < BinaryVec
         PRESCALER <= PRESCALER +1;
      else
         reset PRESCALER to 0;
         COUNTER <= COUNTER +1;
      endif
      if COUNTER = M
         reset COUNTER to 0;
      endif
   end for
```

Figure 5 Algorithm A

```
Algorithm B: Pulse Counter and Heartrate BPM
------------------------------------------------------------
Objective:  Sending heartrate BPM to the seven-segment decoders at
every M=15 seconds, monitor Bradycardia and Tachycardia

Input: Clock, PulseSensor,
Output: PulseLED, TachyLED, BradyLED, NormalLED,
        BPM (8 bits)
internal signal SensorCOUNTER with 8 bits
internal signal Pulse_COUNT15 with 8 bits
  for every tick (10ns) of the Clock
     if  PulseSensor = '1' then
        Pulse_COUNT15 <= Pulse_COUNT15 +1;
        PulseLED <= '1';
     else
        PulseLED <= '0';
     end if;
     Recall algorithm A: M=15 seconds generator
      if   COUNTER = "1111" then
         SensorCOUNTER <= Pulse_COUNT15 logical
                             left shift 2 bits
      else
         SensorCOUNTER <= (others => '0');
      end if
      if SensorCOUNTER > "0110 0100" - - decimal 100
        TachyLED <= '1';
      end if
      if SensorCOUNTER < "0011 1100" - - decimal 60
        BradyLED <= '1';
      end if
      if SENSORCOUNTER) > 60
        and SENSORCOUNTER <100
        NormalLED <='1';
      else
        NormalLED <='0';
      end if
        BMP <= SensorCOUNTER;
        1st digit of 7 segment decoder <= BMP (3 down to 0);
        2nd digit of 7 segment decoder <= BMP (7 down to 4);
  end for
```

Figure 6 Algorithm B

Pulse sensor FPGA peripheral interface test and analysis

The pulse sensor peripheral interface has two logical inputs, four logical outputs, and three vectors outputs (Figure 7). The test is conducted on the Nexys 4 FPGA board and the vector waveform simulation. A 100MHz signal is derived from the system clock 450MHz and will be sent to the input line Clock. Pulse sensor hardware connects to digital logic input line Pulse Sensor that sensing a heartbeat. Beat LED will connect to one LED light on the Nexys 4 FPGA board to indicate the peak of the pulse. When real time elapses a 15-second, the internal signal variable COUNTER reaches 15 and SensorCOUNTER is reset to zero. This process will be repeated by resetting COUNTER to 0. The lower nibble and upper nibble of SensorCOUNTER will be outputted to the inputs of seven segment decoders. Hex1BPM and Hex1BPM will connect to two seven-segment displays to display the updated pulse reading BPM every 15seconds. The digital output line Tachy LED will turn on a LED when BPM is greater than 100heartbeats. The digital line Brady LED will turn on a LED to indicate BPM less than 60heartbeats. When the heartbeat rate is normal, the digital line of Normal LED will show logical high to turn on a LED.

```
-- Pulse sensor peripheral interface
ENTITY PulseInterface is
 Port (
 Clock : in STD_LOGIC;
 PulseSensor : in STD_LOGIC;
 BeatLED : out STD_LOGIC;
 TachyLED : out STD_LOGIC;
 BradyLED : out STD_LOGIC;
 NormalLED : out STD_LOGIC;
 BPM : out STD_LOGIC_VECTOR(7 downto 0);
 Hex1BPM:out STD_LOGIC_VECTOR(6 downto 0);
 Hex2BPM: out STD_LOGIC_VECTOR(6 downto 0));
END PulseCounter;
```

Figure 7 Pulse sensor FPGA peripheral interface written by VHDL.

Clock = 100 MHz can be converted to the binary number Binary Vec = 101111101011110000100000000 in digital logic design. The PRESCALER is set to 0 initially and increased by 1 at each clock tick. After the Clock clicks 100million times, the PRESCALER reaches the binary number 101111101011110000011111111, and a 1-second timer is generated. With the input pulse Clock= 100MHz, the period of each clock is 10ns. The simulated waveform (Figure 8) shows the designed FPGA peripheral interface line. Beat LED can respond to the input signal Pulse Sensor in 15nanoseconds. This responding time can be reconfigured by changing the Clock=100MHz to different frequencies. For example, if the digital system adds n=4 bits frequency divider, the clock= 100MHz can be divided to obtain the output frequency 100MHz÷2^4 =6.25MHz. The period of each clock clicking is reconfigured to 160ns. The input interface line Beat LED will respond to the input pulse signal Pulse Sensor at 160ns approximately (Figure 9). Our implemented FPGA based IoT low-end peripheral interface for the health monitoring environment can distinguish between user normal, Bradycardia, or Tachycardia heart rate (Figure 10). The vector waveform shows that Tachy LED displays a logical high when BPM is 112 or 128. The digital line Brady LED has logical high output when BPM is 48 or 32. The digital line of Normal LED will show a logical high when BPM is 64, 80, or 96. The scenario of the FPGA pulse sensor peripheral integrated with other interface peripherals, Raspberry Pi, Bluetooth, Wi-Fi, and Firebase in our project is shown in Figure 11. Demo of FPGA based pulse sensor health monitoring platform is shown in Figure 12. The pulse sensor is amped in close contact with the fingertip, and the green light on the FPGA board indicate user pulse is beating.

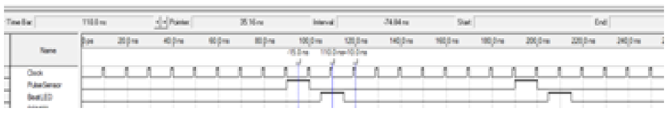

Figure 8 FPGA peripheral output line BeatLED vs. input signal PulseSensor under clock tick 10ns.

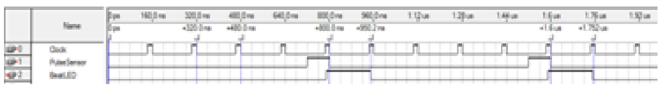

Figure 9 FPGA peripheral output line BeatLED vs. input signal PulseSensor under clock tick 160ns.

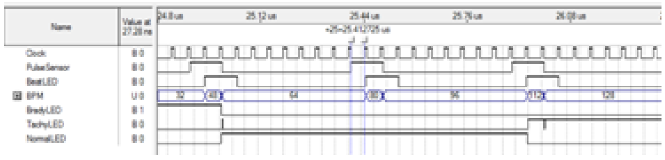

Figure 10 FPGA peripheral to distinguish between user normal, Bradycardia, or Tachycardia.

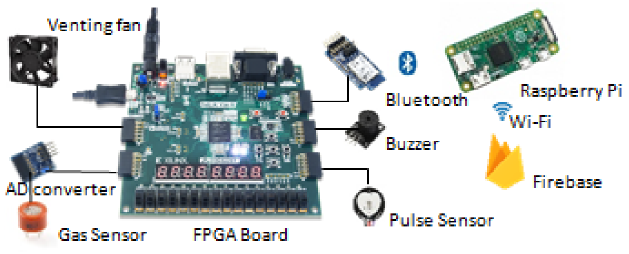

Figure 11 The scenario of the FPGA based IoT low-end sensors and actuators in our project.

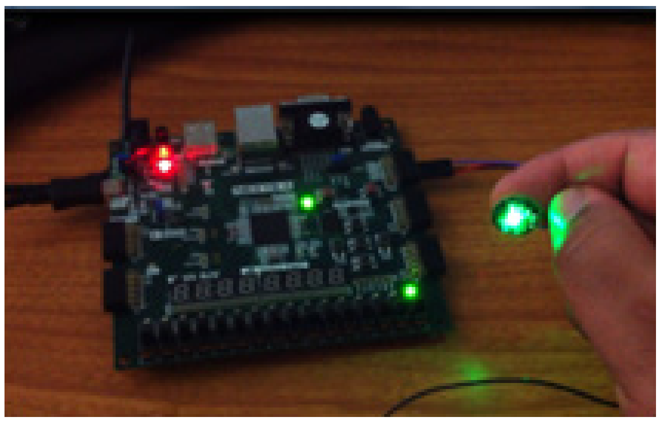

Figure 12 Heartbeats are measured at our FPGA based pulse sensor interface.

Conclusion and future works

An IoT–based smart healthcare network can be built efficiently with FPGA devices. The FPGA-based platform is an inexpensive early validation platform. The sensor interface peripheral implemented by the FPGA technology is reconfigurable, requires less power consumption, and has more accuracy. Future work includes connecting the FPGA based low-end peripheral to our platform such as the Raspberry Pi, Bluetooth, Wi-Fi, and Firebase to monitor human heartbeats and relay user's normal, Bradycardia, or Tachycardia heart rate back to caregivers and other medical devices via IoT technology. The FPGA based gas sensor peripheral interface will be implemented in the future to detect harmful gases. A fan and beeper will integrate into the project to exhaust the harmful gases out of the closed space and send an alarm to a caregiver or relatives via IoT.

Acknowledgments

This research is supported by PSC-CUNY Research Awards 60310-0048 and 61463-0049.

Conflicts of interest

Authors declare that there is no conflicts of interest.

References

1. Hahm O, Baccelli E, Petersen H, et al. Operating Systems for Low-End Devices in the Internet of Things: A Survey. *IEEE Internet of Things Journal*. 2016;3(5):720–734.
2. Grgić K, Zagar D, Križanović V. Medical applications of wireless sensor networks - current status and future directions. *Medicinski Glasnik*. 2012;9(1):23–31.
3. Gomes T, Salgado F, Pinto S, et al. A 6LoWPAN Accelerator for Internet of Things Endpoint Devices. *IEEE Internet of Things Journal*. 2018;5(1):371–377.
4. Internet of Things - number of connected devices worldwide 2015-2025.
5. Chen D, Cong J, Gurumani S, et al. Platform Choices and Design Demands for IoT Platforms: Cost, Power and Performance Tradeoffs. *IET Cyber-Physical Systems: Theory & Applications*. 2016;1(1):70–77.
6. Yamaguchi S, Miyazaki T, Kitamichi J, et al. Programmable wireless sensor node featuring low-power FPGA and microcontroller. *2013 International Joint Conference on Awareness Science and Technology & Ubi-Media Computing (iCAST 2013 & UMEDIA 2013)*. 2013;596–601.
7. A de la Piedra, Braeken A, Touhafi A, Sensor Systems Based on FPGAs and Their Applications: A Survey. *Sensors*. 2012.
8. Gomes T, Pinto S, Gomes T, et al. Towards an FPGA-based edge device for the Internet of Things. *2015 IEEE 20th Conference on Emerging Technologies & Factory Automation*. Luxembourg. 2015:1–4.
9. Chi Q, Yan H, Zhang C, et al. A Reconfigurable Smart Sensor Interface for Industrial WSN in IoT Environment. *IEEE Transactions on Industrial Informatics*. 2014;10(2):1417–1425.
10. Rizal A, Riyadi MA, Darjat. FPGA-based system for continuous monitoring of three vital signs of human body. *2015 2nd International Conference on Information Technology, Computer, and Electrical Engineering (ICITACEE)*. Semarang. 2015;221–226.
11. Mattada MP, Magadum SM, Guhilot H. Identification of hotspots on FPGA using Time to Digital Converter and distributed tiny sensors. *2015 2nd International Symposium on Physics and Technology of Sensors (ISPTS)*. Pune. 2015;235–239.
12. Elsayed M, Badawy A, Mahmuddin M, et al. FPGA implementation of DWT EEG data compression for wireless body sensor networks," *2016 IEEE Conference on Wireless Sensors (ICWiSE)*. Langkawi. 2016;21–25.

13. Yu Wang, Hunter W, Chen X, et al. Improved Hardware Design of IoT Prosthetic Device. The 2018 ASEE Mid-Atlantic Fall Conference. Brooklyn, NY. 2018:26–27.
14. Yu Wang, Sunghoon Jang. Lab Manual Design with Engineering Learning Style and Flipped Learning Model in Computer Engineering Technology Education. 2018 ASEE Northeast Section Annual Conference. Hartford, CT, April 27-28. 2018.
15. https://reference.digilentinc.com
16. https://pulsesensor.com/pages/open-hardware
17. https://www.broadcom.com
18. https://www.kingbrightusa.com/
19. https://www.microchip.com/