Hardware Design: Wireless Electro-cardiogram Monitor

Submitted as a requirement
for the course ICOM5047: Section 030
Design Course for B.S. in Computer Engineering Department
University of Puerto Rico at Mayagüez, PR

Students:

Bermúdez Soto, Wilfredo

Ortiz Pérez, Alexis M.

Vega Diaz, Rafael

Rivera Suarez, Melvin

Faculty Advisors:

José Fernando Vega, PhD
Associate Professor

Nayda G. Santiago Santiago, PhD
Associate Professor

Kejie Lu, PhD
Assistant Professor

Date: December 9, 2008

Hardware Design Document

Table of content

1 Introduction

This document contains all specifications for the hardware components in the WARM ECG Monitoring System. It contains part specifications, implementation, and a firmware design consideration.

2 Hardware Design Specifications

This section describes the hardware design in details.

2.1Components

Our system's hardware is composed of the following parts: microcontroller msp430F149, Bluetooth module Bluegiga WT12, and an analog circuit to process the ECG signal. Figure 1 present a diagram of the main components and their interaction.

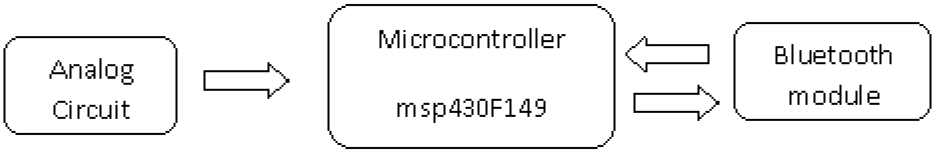

Figure 1 - Block diagram of component interaction

Schematic on figure 2 presents the full system with all connections. For simplicity, the grounds and supply voltage were labeled and not connected using wires.

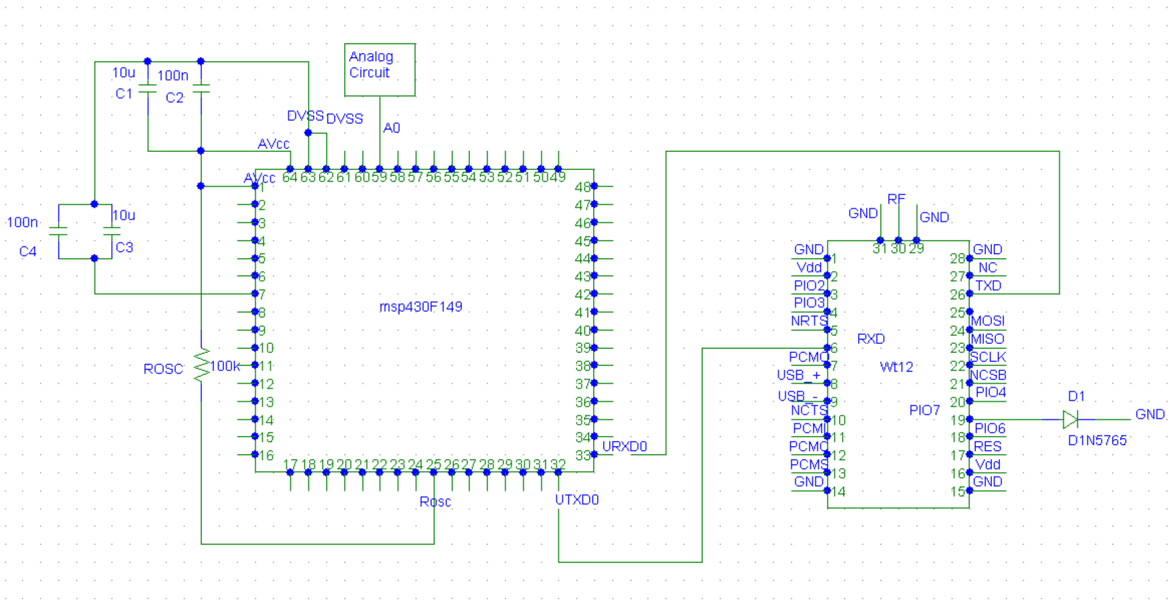

Figure 2 - System schematic. Shows the connections between the devices

2.2 Design Specification

This section describes the components, why we select it and the power consumption

2.2.1Microcontroller msp430F149

It was selected considering functionality, performance, size, cost, and power consumption. Our design requires us to send the data over to a monitoring device. This means that we need to interface the MCU with a wireless communication module. This microprocessor provides an UART interface for that purpose. To make our device wearable and portable, the size of the MCU must be as small as possible. It can support clock rates up to 8MHz. It has 60 KB of flash ROM for code, and 2 KB of RAM for data. It also has two 16-bit timers, a 12-bit ADC with internal reference, sample and hold, auto scan feature.

Power Consumption

It is an ultra low power consumption microcontroller [4]:

- Supply-Voltage 3.3 V
- Active Mode: 280 µA

2.2.2 Bluegiga wt12 Bluetooth module

This module is a class 2, Bluetooth® 2.0+EDR (Enhanced Data Rates) module. It has an approximate range of 10 meters and is a highly integrated module with a build in antenna. Its dimensions are 14mm x 25mm, making it small enough to integrate to the portable system. WT12 is equipped with iWrap firmware, an easy to use firmware that provides a functional access to Bluetooth with ASCII commands send via serial UART interface [1].

Power Consumption

- Supply Voltage 3.3V
- Connect – 22.4 mA, no data was transmitted.
- At TX or RX mode- 70 mA.

2.2.3 Analog Circuit

This analog circuit is a previous design by Chia-HungChen, Shi-Gun Pan and Peter Kinget of the Columbia Integrate System Laboratory[]. The circuit was slightly modified to fit our device’s needs.

- AD624 - The AD624 is a high precision, low noise, instrumentation amplifier with high gain accuracy, low gain temperature coefficient and high linearity used in high resolution data acquisition systems. It offers input noise levels below 4 nV/√Hz at 1 kHz and pin-programmable gains of 1, 100, 200, 500 & 1000 provided on the chip. This amplifier also has a

CMRR above 80 dB at unity gain, which helps reject other common mode signals the electrodes will pick up.

- PS2506 - To provide electrical isolation between the patient and the rest of the device, an opto coupler was used. The PS2506 opto coupler is optically coupled isolators containing a light emitting diode and an NPN silicon Darlington connected phototransistor. It was selected due to its low cost and availability. To electrically insulate the AD624 output signal from the rest of the circuit, it is connected to the output of the AD624 and the output signal is then fed to the MCU's internal analog to digital converter (ADC). The internal circuit and the external terminal unit are electrically isolated from the measurement electrode of the electrocardiograph to prevent the occurrence of an electric shock accident.

- UA741 – This 741-Opamp is used to offset the output DC voltage by connecting it to the reference node of the AD624. The 741 is connected as a buffer and provides low impedance at the reference terminal. This helps keep the CMRR at the desired level, since impedance at this terminal reduces the CMRR by 10 $k\Omega/R_{REF.}$ It can also provide some electric safety, because the auxiliary op amp will saturate when an abnormally high voltage appears between the patient and ground. The UA741 was selected because of its initial availability.

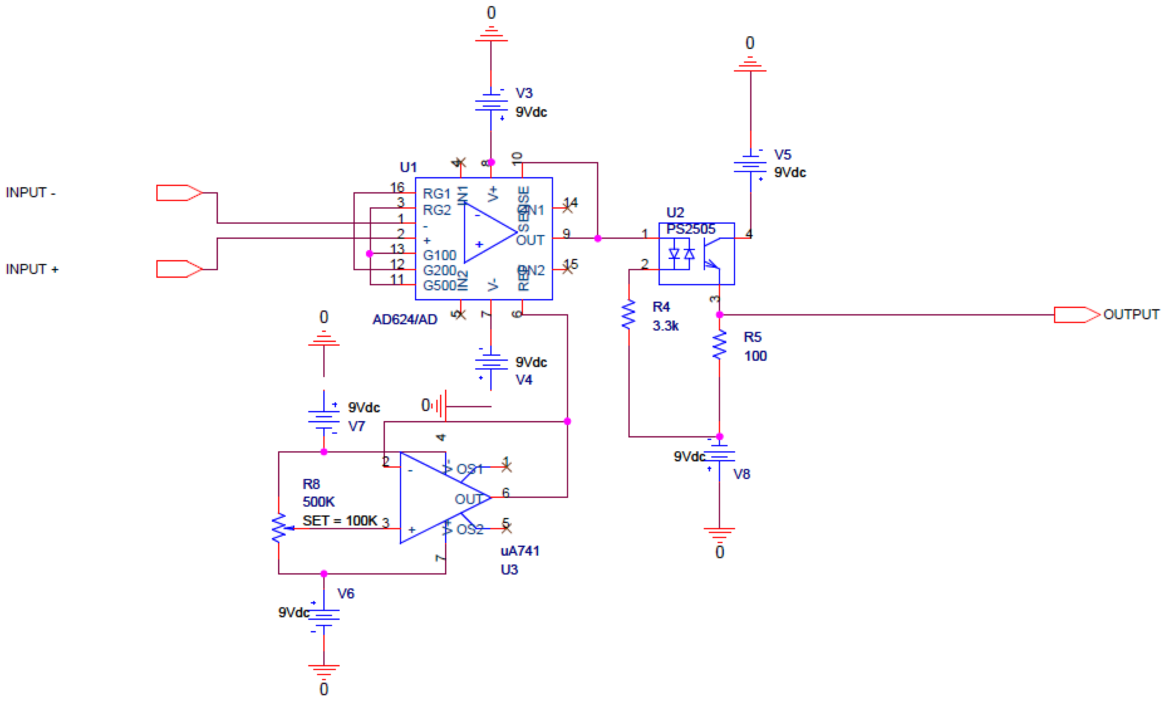

Figure 3 - ECG Circuit

2.2.4 Total power consumption

This section presents the power consumption of the system. For our project the analog power consumption wasn’t taken into consideration because analog circuit design was beyond the scope of our project.

-Tx or Rx mode

This is the power consumption when an alert message or data is send to cell and when data is received to introduce doctors parameters.

3.3*(70 mA + 280 µA) = 231.9 mW

- No data was transmitted

When the circuit isn’t receiving or transmitting data between microcontroller and Bluetooth

3.3*(22.4mA + 280 μA) = 74.8 mA

2.3 Implementation details

This section describes the details of the hardware implementation of the microcontroller and Bluetooth module, including their connections.

2.3.1 MSP430F149

This microcontroller provides UART, timer and ADC integrated peripheral used in our design.

- UART – this peripheral is used to communicate with the Bluetooth module. It is configured at a baud Rate of 9600 bit per second, with 8 data bits, no parity bit, non hardware flow control and one stop bit.

- ADC - ADC peripheral is a 12 bit resolution analog to digital converter. It is used to convert the analog circuit output signal. The sample rate was set to 360 samples per second in order to satisfy the requirements of the firmware algorithm on microprocessor, explained on the firmware section on this document. To reach that sample time, a timer was configured to control the interrupt of the ADC converter. This peripheral needs a reference voltage in order to convert the signal. This reference voltage was set using the internal reference voltage to a range of 0-2.5 volts. To use these internal references and reach high resolution on samples, 4 capacitors were connected in the following way to eliminate ground loops, unwanted parasitic effects, and noise: One 10 uF capacitor in parallel with one 100 nF capacitor connected between positive reference (pin 7 of microcontroller) and ground, and a 10 uF capacitor in parallel with one 100 nF capacitor connected between supply voltage (pin 1) and ground (pin 64). For detail see figure 2.

- Timer - Timer A is used in our system to get the required sample rate of 360. It produces the input clock of the ADC. The timer is configured on a toggle/reset mode. This means that the output is toggled when the timer counts to the TACCR1 (control register) value. It is reset when the timer counts to the TACCR0 (control register) value.

2.3.2 Bluetooth module

This part describes the Bluetooth implementation details including the pin connection consideration. It contain an integrate firmware the iWrap that enables users to access Bluetooth® functionality with simple ASCII commands delivered to the module over the UART interface. The following configuration was made on our system:

- Serial port profile
- UART baud rate of 9600, 8 bits and 1 stop bit.
- Programmed pass code to 0000, it is the password to connect with other devices.

2.3.3 Analog Circuit

The ECG signal we wish to amplify has amplitude of approximately 1mV-5mV. For the original circuit, the AD624 was programmed with a gain of 1000 to bring the signal up to the order of volts. However, due to the limitations in voltage swing set by the ADC, the gain had to be reduced to 500. Also, the offset had to be adjusted to set the signal inside the ADC's range. Initially, it was set to a level of 1V, but it has to be adjusted depending on the patient's waveform.

3 Firmware

This section describes de firmware of the system. The different considerations and limitations are also part of this section.

3.1 Firmware consideration

Due to the microprocessor limitation of 2K bytes memory and 8MHz speed the firmware of WARM ECG requires a fast algorithm and that uses the least possible memory space. By requirement of the system, it was designed and optimized to run at an approximate 25ms, enough to sample two wave forms. That is because the detection of anomalies has to be very fast in order to reach a solution, to approximate a real time device. To reach that resolution, we have to do all division and multiplication using only shifts in order to use all the microprocessor architecture features. Other consideration is to move all needed data that was on the flash memory to the RAM memory in the initialization process and also not write on flash during the process because flash memory access consume approximately 10 cycles of microprocessor clock. That's high compared with a clock RAM access.

3.2 Modules and details

This section presents the different modules and a description.

3.2.1 UART

- *__interrupt void usart0_rx (void)* – Interrupt for the Rx port. It stores the received data from the Bluetooth module on receive data array.
- *void UART_Initialize( )* - initialize UART to a baud rate of 9600 bit per seconds, 8 bits and 1 stop bit. It also enables the interrupt.
- *void UART0_putchar (char c)* - sends the char received as parameter via the UART Tx.

- *void send_to_UART (char* c)* – it receives a pointer to an array of characters and sends it via UART. It uses the UART0_putchar module.
- *void parameter_Append( )* – sends patient parameters to the Bluetooth.
- *void emergency_Alert( )* – sends alert message to the Bluetooth module. This message is explained on the communication protocol appendix.
- *void command Parse(char* c)* – translates the message received from the Bluetooth module

3.2.2 *ADC (analog to digital converter*)

- *void ADC_Initialize ( )* – initializes the ADC converter peripheral. It also configures timer A to be used as it internal clock to reach the 360 samples per seconds rate.
- *void convert_ADC_Enable ( )* – enables the ACD conversion turning on the enable bit on ADCCTL0 register. See msp430 User Guide on ADC12 section.
- *void convert_ADC_Disable ( )* – disables the ADCCTL0 enable conversion bit. More details on msp430 Family User Guide.
- *__interrupt void ADC12ISR ( )* – activates conversion when the sample clock (timer A) send the signal and save the converted sample on dataValue array.

3.2.3 Flash memory

- *void write_to_flash(int save, int* direction)* – saves an integer received as parameter on the direction received by the pointer parameter.
- *void Flash_clr(float *Data_ptr)* – removes the data stored on flash segment pointed to by the received as parameter.
- *void saveBaseParameters(char* c, int flag)* – received a pointer to the string with the received parameter to save on flash and a flag to

specify if the received parameters are lower bound or high bound. With this parameter it makes the decision of what address the data will be saved.

3.2.4 Peak detection

- *void look2future(int *dataValue ,int zero, int dataSize, int look2future, int recount, int* peakPosition, int* peakLeftBound, int* peakRightBound)* – this function receives the pointer of the data sampled array, the size of the array, and store the position of the peak on the peakPosition array. Also save the position of peak left bound and peak right bound of the detected peak.
- *int left_bound_analysis (int *dataValue, int actual_point) –* receives an array of data and the peak position, then returns the peak left bound position.
- *int right_bound_analysis (int *dataValue, int actual_point)* - receives an array of data and the peak position, then returns the peak right bound position.
- *int lower_value (int *dataValue , int leftLimit, int rightLimit)* - receive a pointer to the array of data we want to and the bounds, to set the interval, and return the lower value of this on this interval.
- *int higher_value (int *dataValue , int leftLimit, int rightLimit)- r*eceives a pointer to the array of data, and the bounds to set the interval, and returns the higher value of this on this interval.

3.2.5 Main

Main function first calls all initialization methods and set the required ports. If it is the first time that the device is activating, it will wait for the doctor to enter the parameters. When the parameters are set, it enters on a loop that takes samples and analyses it. In case that analysis results on a value out of doctor parameter range then it send an alert to the patient, if a request of data is received, then it sends the analysis of the results.

4 References

[1] Bluegiga, "wt12 Data Sheet", Version 2.5, November 16, 2007

[2] Bluegiga, "iWrap 2.2.0 User Guide", version 3.3, November 21, 2007.

[3] Texas Instrument, "msp430F149x1xx User's Guide, 2006.

[4] Texas Instrument, "msp430x13x, msp430x14x, msp230x14x1 Mixed Signal Microcontroller", July 2000- Revised June 2004.

[5] Texas Instrument, "MSP430 Universal Synchronous Asynchronous Receive/Transmit Communication Interface", Application Report, April 1999.