Chinese Traditional Medicine Journal

ISSN : 2693 6356
2020 | Vol 3 | Issue 4

HANDHELD SENSORS FOR USE IN SPORTS BIOMECHANICS

Dr. Abhi [1],V.Balaji[2] , G.Sujatha[3],G.Lakshmi [4]

[1,2,3,4] Associate professor,

Sreenidhi Institute of Science and Technology (SNIST) , Hyderabad,Telangana,India,

ABSTRACT:

Research in the subject of sport biomechanics aims to quantify athlete performance, provide coaches with effective tools and instructions to use in their training, and reduce the likelihood of injury. New technologies have made it possible to analyze athletes' motions in real time, without impeding their performance, by equipping them with tiny, wearable sensors.
Keywords: biomechanics, sports, sport psychology, etc.

1. INTRODUCTION.

These days, technology dictates almost every aspect of human existence. It is vital to meeting the needs of modern society. Using cutting-edge technology, we can improve not just our capacity to keep tabs on people's health, but also their general level of comfort. Here, we're using cutting-edge technology to detect life-threatening medical conditions, all in the name of getting patients the prompt care they need.

The purpose of this study is to create a Bluetooth-based system for continuous health monitoring. Here, we employ MAX30100 sensors to track the patient's heart rate and spo2, and LM35 sensors to monitor core body temperature. Wireless health monitoring system built on the Arduino platform.

Arduino is a user community, open-source hardware and software project, and manufacturer of microcontroller kits for creating electronics and interactive installations.objects that can sense and control objects in the physical world.

The functioning of this device is based on the truth that the blood level circulation during expansion and contraction of heart which can be sensed by Heartbeat sensor. Depending upon the rate of circulation of blood per second the heart beat rate per minute is calculated. This device consists of an Arduino microcontroller which takes the input from the heart beat, temperature sensor and calculates the heart rate, spo2 and temperature of the patient. The micro controller takes the responsibility to sends the sensor data into the bluetooth mobile application using HC-05 Bluetooth module. The Microcontroller is programmed using Embedded C language.

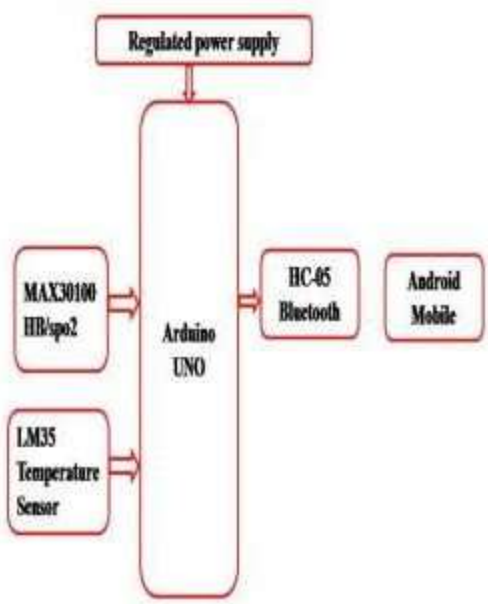
Motivation

An embedded system is a combination of software and hardware to perform a dedicated task. Some of the main devices used in embedded products are Microprocessors and Microcontrollers.

Microprocessors are commonly referred to as general purpose processors as they simply accept the inputs, process it and give the output. In contrast, a microcontroller not only accepts the data as inputs but also manipulates it, interfaces the data with various devices, controls the data and thus finally gives the result.

The Project "**Health Monitoring System**" is an exclusive project that can monitor the patient's health parameters such as body temperature, spo2, heartbeat and sending them into the Bluetooth mobile application.

BLOCK DIAGRAM OF THE PROJECT



The main blocks of this project are:

1. Regulated power supply.
2. ARDUINO Micro controller.
3. MAX30100Heart beat, SPO2 sensor.
4. LM35Temperature sensor.
5. HC-05 Bluetooth module.

Advantages:

- Wireless transmission of medical parameters.
- Real-time health parameters monitoring system.
- Using Bluetooth technology.
- Patient's health status can be continuously monitored.
- Low power consumption.
- Efficient design.
- Higher accuracy.

Disadvantages:

- The system can't work if the sensor fails.

Applications:

We can implement this project in

- HOSPITALS

Methodology

Features

• High Performance, Low Power AVR® 8-Bit Microcontroller

• Advanced RISC Architecture

– 131 Powerful Instructions – Most Single Clock Cycle Execution

– 32 x 8 General Purpose Working Registers

– Fully Static Operation

– Up to 20 MIPS Throughput at 20 MHz

– On-chip 2-cycle Multiplier

• High Endurance Non-volatile Memory Segments

– 4/8/16/32K Bytes of In-System Self-Programmable Flash progam memory (ATmega48PA/88PA/168PA/328P)

– 256/512/512/1K Bytes EEPROM (ATmega48PA/88PA/168PA/328P)

– 512/1K/1K/2K Bytes Internal SRAM (ATmega48PA/88PA/168PA/328P)

– Write/Erase Cycles: 10,000 Flash/100,000 EEPROM

– Data retention: 20 years at 85°C/100 years at 25°C(1)

– Optional Boot Code Section with Independent Lock Bits

In-System Programming by On-chip Boot Program

True Read-While-Write Operation

– Programming Lock for Software Security

• Peripheral Features

– Two 8-bit Timer/Counters with Separate Prescaler and Compare Mode

– One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture

Mode

– Real Time Counter with Separate Oscillator

– Six PWM Channels

– 8-channel 10-bit ADC in TQFP and QFN/MLF package

Temperature Measurement

– 6-channel 10-bit ADC in PDIP Package

Temperature Measurement

– Programmable Serial USART

– Master/Slave SPI Serial Interface

– Byte-oriented 2-wire Serial Interface (Philips I2C compatible)

– Programmable Watchdog Timer with Separate On-chip Oscillator

– On-chip Analog Comparator

– Interrupt and Wake-up on Pin Change

• **Special Microcontroller Features**

– Power-on Reset and Programmable Brown-out Detection

– Internal Calibrated Oscillator

– External and Internal Interrupt Sources

– Six Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, Standby,

and Extended Standby

• I/O and Packages

– 23 Programmable I/O Lines

– 28-pin PDIP, 32-lead TQFP, 28-pad QFN/MLF and 32-pad QFN/MLF

• Operating Voltage:

– 1.8 - 5.5V for ATmega48PA/88PA/168PA/328P

• Temperature Range:

– -40°C to 85°C

• Speed Grade:

– 0 - 20 MHz @ 1.8 - 5.5V

• Low Power Consumption at 1 MHz, 1.8V, 25°C for ATmega48PA/88PA/168PA/328P:

– Active Mode: 0.2 mA

– Power-down Mode: 0.1 µA

– Power-save Mode: 0.75 µA (Including 32 kHz RTC)

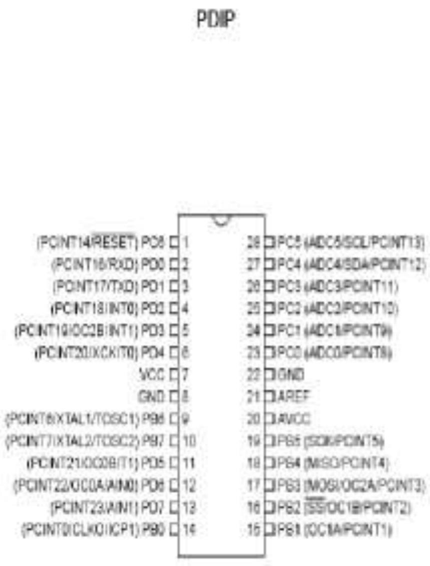
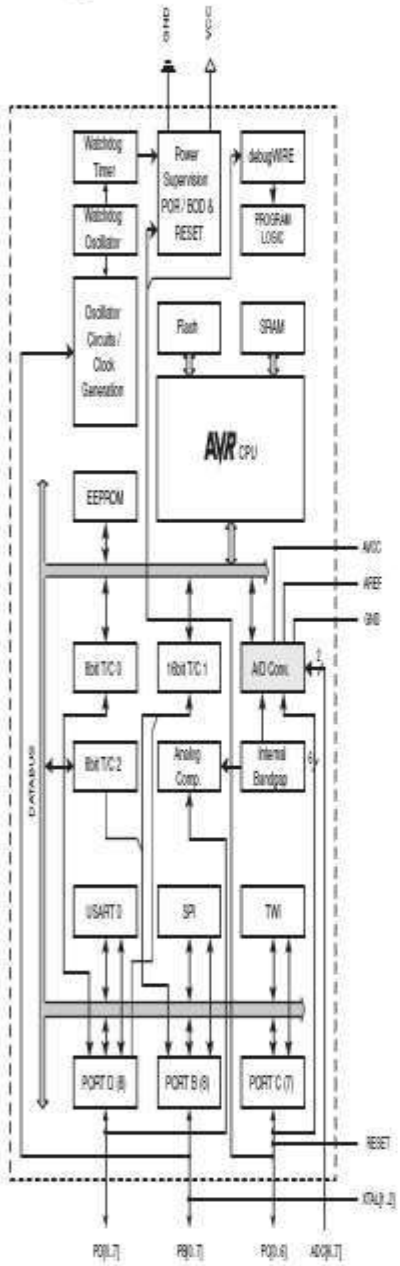


Fig:

Overview

The ATmega48PA/88PA/168PA/328P is a low-power CMOS 8-bit microcontroller based on the AVR enhanced RISC architecture. By executing powerful instructions in a single clock cycle, the ATmega48PA/88PA/168PA/328P achieves throughputs approaching 1 MIPS per MHz allowing the system designer to optimize power consumption versus processing speed.

Figure 2-1. Block Diagram



The AVR core combines a rich instruction set with 32 general purpose working registers. All the 32 registers are directly connected to the Arithmetic Logic Unit (ALU), allowing two independent registers to be accessed in one single instruction executed in one clock cycle. The resulting architecture is more code efficient while achieving throughputs up to ten times faster than conventional CISC microcontrollers.

The ATmega48PA/88PA/168PA/328P provides the following features: 4K/8K bytes of In-System Programmable Flash with Read-While-Write capabilities, 256/512/512/1K bytes EEPROM, 512/1K/1K/2K bytes SRAM, 23 general purpose I/O lines, 32 general purpose working registers, three flexible Timer/Counters with compare modes, internal and external interrupts, a serial programmable USART, a byte-oriented 2-wire Serial Interface, an SPI serial port, a 6-channel 10-bit ADC (8 channels in TQFP and QFN/MLF packages), a programmable Watchdog Timer with internal Oscillator, and five software selectable power saving modes. The Idle mode stops the CPU while allowing the SRAM, Timer/Counters, USART, 2-wire Serial Interface, SPI port, and interrupt system to continue functioning. The Power-down mode saves the register contents but freezes the Oscillator, disabling all other chip functions until the next interrupt or hardware reset.

In Power-save mode, the asynchronous timer continues to run, allowing the user to maintain a timer base while the rest of the device is sleeping. The ADC Noise Reduction mode stops the CPU and all I/O modules except asynchronous timer and ADC, to minimize switching noise during ADC conversions. In Standby mode, the crystal/resonator Oscillator is running while the rest of the device is sleeping. This allows very fast start-up combined with low power consumption. The device is manufactured using Atmel's high density non-volatile memory technology. The On-chip ISP Flash allows the program memory to be reprogrammed In-System through an SPI serial interface, by a conventional non-volatile memory programmer, or by an On-chip Boot program running on the AVR core. The Boot program can use any interface to download the application program in the Application Flash memory. Software in the Boot Flash section will continue to run while the Application Flash section is updated, providing true Read-While-Write operation. By combining an 8-bit RISC CPU with In-System Self-Programmable Flash on a monolithic chip, the Atmel ATmega48PA/88PA/168PA/328P is a powerful microcontroller that provides a highly flexible and cost effective solution to many embedded control applications.

The ATmega48PA/88PA/168PA/328P AVR is supported with a full suite of program and system development tools including: C Compilers, Macro Assemblers, Program Debugger/Simulators, In-Circuit Emulators, and Evaluation kits.

Comparison Between ATmega48PA, ATmega88PA, ATmega168PA and ATmega328P .

Table 2-1. Memory Size Summary

Device	Flash	EEPROM	RAM	Interrupt Vector Size
ATmega48PA	4K Bytes	256 Bytes	512 Bytes	1 instruction word/vector
ATmega88PA	8K Bytes	512 Bytes	1K Bytes	1 instruction word/vector
ATmega168PA	16K Bytes	512 Bytes	1K Bytes	2 instruction words/vector
ATmega328P	32K Bytes	1K Bytes	2K Bytes	2 instruction words/vector

The ATmega48PA, ATmega88PA, ATmega168PA and ATmega328P differ only in memory sizes, boot loader support, and interrupt vector sizes. Table 2-1 summarizes the different memory and interrupt vector sizes for the three devices.

ATmega88PA, ATmega168PA and ATmega328P support a real Read-While-Write Self-Programming mechanism. There is a separate Boot Loader Section, and the SPM instruction can only execute from there. In ATmega48PA, there is no Read-While-Write support and no separate Boot Loader Section. The SPM instruction can execute from the entire Flash.

Implimentation

The project "**Health Monitoring System"** was designed a continuous patient health monitoring system using Bluetooth technology. This project makes a use of MAX30100 (heartbeat & oxygen), LM35 temperature, Bluetooth module. The main controlling device of the project is Arduino UNO microcontroller. Arduino will continuously read the heartbeat and spo2 value through MAX30100 and temperature value from LM35 temperature sensor will be sent to the user android mobile application via HC-05 bluetooth module. To achieve this task microcontroller loaded program written in embedded C language.

This project is implemented using following software's:

- Arduino IDE Studio Compiler - for compilation part

4.1 Arduino IDE Compiler:

This instructable adds to any of the Arduino on a Breadboard instructables.

1. We need a microcontroller with a pre-loaded Bootloader, or must load your own
2. Not all ATmega328's are equal
(A bootloader, very simply, is a programme that sits on the chip and manages the upload of your sketches onto the chip)
Procedural steps for compilation, simulation and dumping:

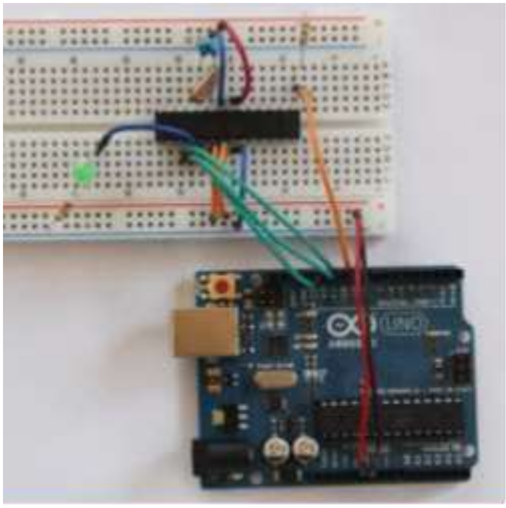
Compilation and simulation steps:

Step 1: Parts

1 x Arduino on a Breadboard
1 x Arduino UNO
Connecting Wires
Arduino IDE installed on your PC

Step 2: The Approach

We use the Arduino UNO to bootload the ATmega328 that is sitting on the Arduino-on-a-Breadboard. This is fairly straightforward having an ATmega328P-PU, but needs an extra step for an ATmega328-PU.



Step 3: Program your Arduino UNO as an ISP

We need to program the Arduino UNO to act as an ISP (In-System Programmer), so that it can burn the bootloader onto the Breadboard chip.

1. Open the Arduino IDE
2. Open the ArduinoISP sketch (under File, Examples)
3. If you're using version 1.0 of the IDE:

   Search for *void heartbeat* and change the line that reads:

   delay(40);

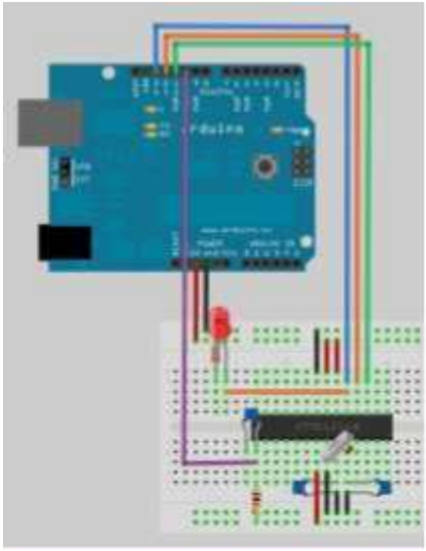
   to

   delay(20);

Connect your UNO to the PC, making sure it's not connected to the Arduino on a Breadboard.

Ensure your UNO is selected under the Boards menu option, and upload the sketch.

Step 4: Connect your ATmega328



Now connect your ATmega to your UNO as follows:

- UNO 5v ---> ATmega pin 7 (VCC)
- UNO GND ---> ATmega pin 8 (GND)
- UNO pin 10 ---> ATmega pin 1 (RESET)
- UNO pin 11 ---> ATmega pin 17 (MOSI)
- UNO pin 12 ---> ATmega pin 18 (MISO)
- UNO pin 13 ---> ATmega pin 19 (SCK)

In your Arduino folder, find the subfolder.*\hardware\tools\avr\etc*

1. Make a backup copy of the file: *avrdude.conf*
2. Open the file *avrdude.conf* in a text editor
3. Search for: “*0x1e 0x95 0x0F*” (this is the ATmega328P signature)
4. Replace it with: “*0x1e 0x95 0x14*” (this is the ATmega328 signature)
5. Save the file
6. Restart the Arduino IDE
7. Continue with the rest of the steps in the instructable, and once bootloading is complete restore the backup copy you made.

In the Arduino IDE, from the *Tools* menu:

- under the *Board* option choose *Arduino UNO*
- under the *Serial Port* option ensure the correct port is selected
- under the *Programmer* option choose *Arduino as ISP*

To burn the Bootloader, choose *Burn Bootloader* from the *Tools* menu
You should see a message *“Burning bootloader to I/O Board (this may take a minute)"*
Once the bootloader has been burned, a message of confirming the success gets displayed.
”**Congratulations: You're now ready to load sketches onto your Arduino on a breadboard!”**

Conclusion:

Its design incorporates functionality from all of the deployed hardware components. Each module's inclusion and placement has been meticulously planned to optimize performance. Second, thanks to developing technology and cutting-edge integrated circuits, the idea has been realized. Therefore, the project's design and testing phases have been fruitful.

References

1. K. Natarajan, B. Prasath, P. Kokila. "Smart Health Care system using Internet of Things." Journal of Network communications and Emerging K. Natarajan, B. Prasath, P. Kokila. "Smart Health Care system using Internet of Things." Journal of Network communications and Emerging Technologyes, 2013: Vol6. Issue 3.

2. Zhang, R.S.H. Istepanaian and T. T. "Guest Editorial introduction the special section: 4G health - the long term evolution of m-health." IEEE Trans Inf. Tech Biomed, 2012: vol.16 no.1 pp 1-5.

3. L. Wang, G.Z. Yang, J. Huang ,J. Zhang, L. Yu, Z. Nie. "A Wireless biomedical signal interface system on chip for body sensors networks"." IEEE Trans. Biomed, Circuits syst, 2010: vol.4No.2 pp112-117.

4. Daeilkwon, Melida, hodkiew. "IoT-Based Prognostics and systems Health Management for Industrial Applications." IEEE Special secion on Trends and Advances for Ambient intellligence with Internet of Things(IoT) systems, 2016.

5. L.Wing. "GE's big bet on data and analytics in proc." MIT Sloan Manage Rev,, 2016: 1-6.

6. D.J.Patil, T.H. Davenport and. "Data Scientist: The Sexiest job of 21st century." Harvard Bus Rec, 2012.

7. Court, D. Barton and D. "Advanced Analytics work for you." Harvard Bus Rev, 2012.

8. M. Gorlatova, J. Sarik, G. Grebla M. Cong, I. Kymissis and G. Zuddman. "Movers and Shakers; Kinetic energy harvesitng for Interntet of Things." IEEE J. Sel Areas Communications, 2015: 1624-1629

9. R. Drath and A. Horch. "Industrie 4.0 Hit or Hype [Industry forumm]." IEEE Ind. Electro . Mag., 2014: Vol 8. No2. PP 56-58.

10. S. cheng, M.H, Azarian and M. Pecht. "Sensor system for Prognostics and health Mangement." Sensors, 2010: vol.10 no.6