RESEARCH ARTICLE

BULLETIN OF SCIENTIFIC RESEARCH

Asian Research Association

DOI: 10.54392/bsr2514

Cost-Effective Smart Gloves for the Paralyzed Hand Rehabilitation

K. Priyadarshan [1,*], F. Spurgeon Jayakaran [1], A. Jayashree [1]

[1] Department of Biomedical Engineering, Bharath Institute of Higher Education and Research, Chennai, India
* Corresponding author Email: priyanprince378@gmail.com
DOI: https://doi.org/10.54392/bsr2514
Received: 28-01-2025; Revised: 20-04-2025; Accepted: 26-04-2025; Published: 07-05-2025

Abstract: Stroke is a severe, frequent, and certain as an international health problem worldwide. Stroke is among most frequent cause of mortality and one of the leading causes of disability in adults. Even with all of the amazing progress made in stroke care, the majority of stroke patients will still require PT intervention if proper care is not provided. This study aims to describe a new, improved gadget that helps stroke patients who are unable to move their hands. The proposed prototype is an accessible smart glove designed to help stroke survivors recover by continuously contracting and relaxing their muscles without the need for physiotherapy. In this project Temperature and Heartbeat sensors are interfaced to microcontroller. Vibration motor will be turned ON to activate hand movement for long time when MEMS sensor remains constant.

Keywords: Physiotherapy Involvement, Stroke Rehabilitation, Smart Glove, MEMS sensor

1. Introduction

According to a WHO survey conducted between 250 000 to 500 000 persons worldwide experience spinal cord injuries (SCI) each year. Most spinal cord injuries result from avoidable causes such violent crimes, falls, and auto accidents. Males are particularly vulnerable when they are young adults (20–29 years old) and older (70+). Adolescence (15–19) and elder adulthood (60+) are the riskiest times for women. According to reports, adult male-to-female ratios are at least 2:1 and sometimes higher. This patient is unable to talk or do things like turn on a fan or drink like other individuals. After considering this problem, we propose a device known as a "smart glove" that will incorporate a feature to assist the disabled and enable them to interact freely with the outside world. If they need assistance, they can call their doctor or leave a message for their caregiver. Therefore, even while these restrictions are beneficial for people with typically able personalities, they won't be very helpful for people who are physically disabled or elderly, as they cannot exert enough tension to move their hands. The objective of this work is to create a therapeutic glove that will help stroke victims recover at home.

This leads to a better recovery rate than the other conventional approaches and decreases the cost of rehabilitation. It can be argued that one of the most serious and certain as an international health concern all over the world. Furthermore, therapy is necessary for stroke survivors to properly recover. It has also been discovered that well-integrated multidisciplinary stroke units support rehabilitation. On recovery of the motor for the arm, some of the options of which may have benefit include constraint-induced movement therapy and robotics [1].

1.1 Embedded system

A computer system is said to be embedded if its software and hardware are so tightly interwoven that it is impossible to see them without compromising the integrity of the two. In others, embedded systems exist as a complete system of its own.

There is one category of embedded processors that concentrates on size, power, and cost. Because of this, certain embedded processors have limited capability; that is, a processor that is suitable for the application category for which it is designed but will inevitably be inappropriate for other application categories. Real-time systems are ones where both timing and functional correctness are necessary for the system's overall accuracy. The accuracy of timing is just as important as the accuracy of function.

1.2 Application of the embedded system

Many embedded systems are used in everyday life, such as in home applications (washing machine, micro oven, security system, MP3 player, DVD, etc.). missiles, automobiles, nuclear research, aircraft, and personal use (iPod, cell phone).

1.3 Embedded system network application

Embedded systems exist to perform some particular function, as opposed to being a generic computer to execute many tasks. For reasons like safety and usability, some of them also have real-time performance requirements that must be met; others have little to no performance requirements, therefore the system hardware is kept as simple as possible to reduce costs. Not all embedded systems are standalone gadgets. The majority are physically built into the gadgets they manage. Firmware, as it is widely known, is the code that is programmed for embedded systems and is stored in read-only memory or Flash memory chips rather than a disk drive. It often runs on a computer with little to no hardware, including a keyboard, a display screen, and memory [2].

2. System Analysis

2.1 Existing System

2.1.1 Physiotherapy

The previous Physiotherapy is performed by a specialist therapist for hand rehabilitation. Hand physiotherapy, or hand therapy, is a specialized area of physiotherapy that is concerned with the assessment and treatment of conditions and injuries of the hand, wrist, and forearm. The human hand is a highly dexterous and very complex body part, and hand physiotherapists have a key function in treating patients with various ailments, from widespread complaints such as carpal tunnel syndrome and tennis elbow to more severe problems encompassing fractures, ligament strains, or rehabilitation following surgery. Hand physiotherapists utilize a multitude of techniques, from manual therapy to therapeutic exercises and modalities like ultrasound and heat to enhance range of motion, hand strength, and function. Splints and braces, designed specifically for each patient, are frequently incorporated into the treatment regimen to assist the patient in restoring maximum hand function and minimizing pain.

The objectives of hand physiotherapy lie beyond symptomatic relief of pain and functional gain. Hand therapists are trained in the treatment of scar tissue, edema (swelling), and sensory loss that may occur due to hand injury or surgery. They collaborate with patients to regain fine motor control and dexterity, essential for daily activities and job-related functions. They also educate patients on hand care and prevention of injury, enabling them to achieve long-term hand health. Whether assisting a person in recovering from a traumatic hand injury or dealing with chronic conditions such as arthritis, hand physiotherapy plays an important role in improving the quality of life and autonomy of hand- related patients.

2.2 Proposed System

2.2.1 MEMS Based Technology

A wearable device called Smart Glove helps stroke survivors recover by constantly tightening and relaxing their muscles, which helps them regain muscle memory. It requires the patient to move their fingers back and forth in order to recover muscle memory and arm functionality. The Smart Glove helps patients exercise on schedule and gives them instant feedback so they can evaluate their performance with the least amount of effort.

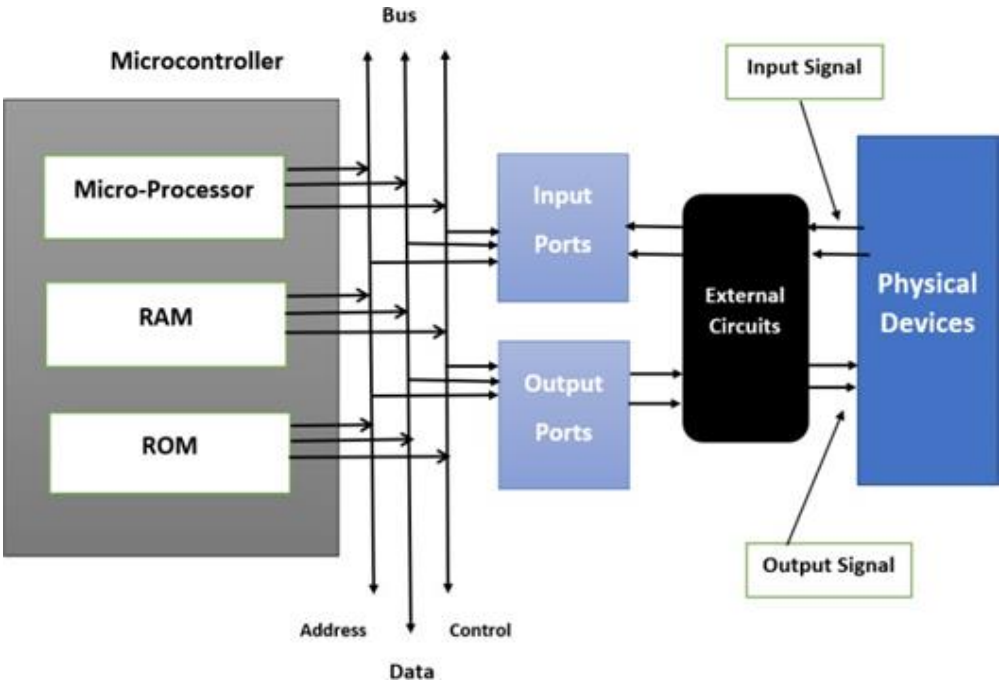


Figure 1. Embedded System Design

It serves as a positioning help for people who have modest hand weakness or muscle stiffness. By allowing the fingers to be in an extended, open hand position with adjustable pressure straps, the extender helps to decrease muscle tone. It is also possible to invert the finger strapping to provide support for a firm grip when holding an object which is clearly displayed in figure 1. In this project we have used MEMS/Flex sensors with vibrator motor. So, when sensor reading is low vibration will be provided to the muscle [8].

2.3 Working Principle

MEMS sensor, Temperature and Heartbeat sensors are interfaced to micro controller. Heartbeat, Temperature and MEMS readings will be indicated in LCD [9]. Data will be sent to the micro controller if the MEMS reading is abnormal. The motor driver will get instructions from the micro controller to turn ON the vibration motor. Vibration motor will turn ON and provides vibration to the whole palm.

2.4 Block Diagram

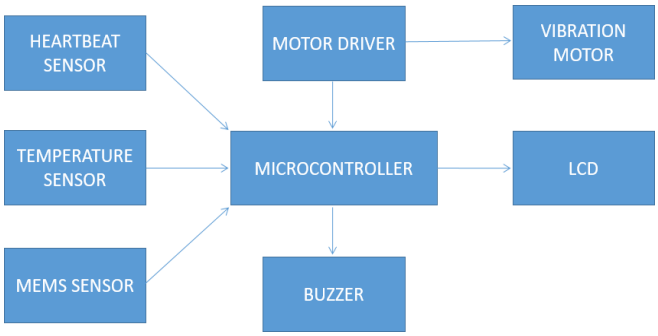


Figure 2. Block diagram

2.5 Project features

- Light weight network.
- Low power consumption.
- High speed networking.
- Broadcast communication

3. Hardware Description

3.1 Arduino Microcontroller

As shown in figure 2, The Arduino Uno is a type of small computer on a circuit board, built with the ATmega328 chip at its core [10]. This board boasts 14 pins for digital inputs and outputs, with 6 of these pins offering PWM output capabilities. This feature is particularly useful for functions like controlling motors or lighting. Additionally, there are 6 analog input pins that are handy for measuring electrical signals. You can connect this board to your computer via a USB cable. Alternatively, it can be powered through an adapter plugged into a wall socket or by a battery. Unlike its predecessors, the Uno does not use the conventional FTDI chip for USB connections. Instead, it utilizes an Atmega16U2 chip (or Atmega8U2 on older R2 versions) configured for USB-to-serial conversion.

3.2 Highlights of Revision 3

The Uno board's third version adds important new features. It improves interoperability with a variety of devices by adding SDA and SCL pins close to the AREF pin [11]. Moreover, the introduction of IOREF and another reserved pin near the RESET pin allows for broader voltage adaptability. This means that the board can function with shields designed for either 5V or 3.3V, making it versatile for both older AVR boards and newer models like the Arduino Due. The board is also equipped with a stronger RESET circuit for improved reliability. Named “Uno,” which is the Italian word for “one,” this board marks the standard platform for Arduino 1.0 and future reference models.

3.3 Schematics & Design Reference

For those interested in the design and layout of the Uno, schematic files are available in EAGLE format. Users can download these files for a closer look at the board’s architecture. Although the current models use the ATmega328 chip, the design allows for backward compatibility with older chips like the Atmega8 and Atmega168 [12]. The pin configuration across these variations remains consistent.

3.4 Power Options for Arduino Uno

Powering the Arduino Uno is highly flexible. You can use a USB connection to your computer or an external source. If you opt for external power, this can be supplied via an adapter or battery. A 2.1mm center-positive connector should be used to connect the adapter to the power jack, and the Gnd and Vin pin headers can be used to connect the battery leads. It is recommended to use a voltage range of 7 to 12 volts for stability. Supplying under 7 volts could lead to instability, while over 12 volts might risk overheating the voltage regulator, potentially damaging the board.

3.5 Details on Power Pins

VIN: This pin accepts the input voltage when using an external power supply instead of USB power. It can provide or receive power depending on configuration.

5V: Uses the board's internal voltage regulator to produce a controlled 5-volts output. The VIN pin, USB connector, or DC power jack are all acceptable power sources. It is crucial not to supply power through

the 5V or 3.3V pins directly; doing so might bypass the regulator and damage the board.

3V3: Supports devices that require a lower voltage by providing a 3.3-volt supply produced by the onboard regulator. 50 mA is the maximum current output.

GND: Several ground pins are available to complete electrical circuits, establishing a common return path for current.

3.6 Memory Capacity

The ATmega328 chip onboard has a total memory capacity of 32 KB, but note that 0.5 KB of this is occupied by the bootloader, which is the start-up firmware. There is also 2 KB of SRAM for active data manipulation and 1 KB of EEPROM for permanent data storage, which can be read from and written to using special EEPROM functions in software.

3.7 Input and Output

There are fourteen digital pins on the Arduino Uno board that can be used for input or output. The pinMode(), digitalWrite(), and digitalRead() routines are used to control them. They run on 5 volts. Each pin can either give or take in up to 40 mA (milliamps) and has a pull-up resistor built in (20-50 kOhms), which is usually off. Some pins have special jobs. Serial Communication: Pins 0 (RX) and 1 (TX) are for receiving and sending serial data. They connect to a chip called ATmega8U2, which helps the USB talk to the board. External Interrupts: Pins 2 and 3 can detect signal changes and trigger actions. They can respond to low or high signals, or when signals rise or fall. Check the attachInterrupt () function to learn more. PWM Output: Analog Write () can generate 8-bit PWM signals from Pins 3, 5, 6, 9, 10, and 11. SPI Communication: SPI communication is managed via pins 10 (SS), 11 (MOSI), 12 (MISO), and 13 (SCK). To utilize them, the SPI library is required. Integrated LED: There is an LED on pin 13. The LED is on when the pin is HIGH and off when it is LOW. Six analog input pins (A0 through A5) with 10-bit resolution (1024 values) are also present on the board. They measure between 0 and 5 volts, however you may adjust this by using analog Reference () and the AREF pin. Certain analog pins serve specific purposes:

TWI Communication: TWI communication with the Wire library is accomplished via pins A4 (SDA) and A5 (SCL). Additional Pins: For analog inputs, AREF provides a reference voltage. If you bring the Reset pin low, the board can be restarted. For Atmega8, 168, and 328 chips, digital pins correspond to ATmega328 ports in the same manner.

3.8 Communication

Computers, other Arduinos and microcontrollers can all communicate with the Arduino Uno. Pins 0 (RX) and 1 (TX) on the ATmega328 allow UART TTL serial communication at 5 volts. The board's ATmega16U2 chip transmits this data over USB, which appears on your computer as a virtual COM port and usually doesn't require an additional driver. You will need a .inf file on Windows. Text data can be sent and received using the serial monitor built into the Arduino program [13]. When data is transferred between the board and USB, the RX and TX LEDs light; pins 0 and 1 are not used for communication. Any digital pin can be used for serial communication with the help of a SoftwareSerial library. SPI and I2C are also supported by the ATmega328. Use the Wire library for I2C and the SPI library for SPI communication [3].

3.9 Programming

Download the Arduino software in order to program the Arduino Uno. Depending on the microcontroller on your board, choose "Arduino Uno" from the Tools > Board menu. Reference materials and tutorials are available for more guidance. The ATmega328 has a pre- installed bootloader, so you can upload new code without extra hardware. It uses the STK500 protocol [14]. You can also program the chip directly through the ICSP header if needed. The ATmega16U2 (or 8U2 on older models) firmware is available. These chips have a DFU bootloader for uploading new firmware:

- On Rev1 boards, activate DFU mode by connecting a solder jumper and resetting the 8U2.
- On Rev2 and later boards, a resistor helps ground the HWB line, making DFU mode easier.

The DFU programmer can be used on Mac OS X and Linux or Atmel's FLIP on Windows to upgrade the firmware. As an alternative, utilize an external programmer and the ISP header. Check user tutorials for more information [4].

3.10 Automatic (Software) Reset

The Arduino Uno can reset itself through software on a connected computer, eliminating the need to press the reset button before uploading new code. This occurs because a tiny capacitor connects the ATmega8U2/16U2's DTR line, a hardware component of the Uno, to the ATmega328's reset line. When the DTR line is activated, it briefly lowers the reset line, causing the chip to reset. With this feature, uploading code is as easy as clicking the Arduino software's upload button. The bootloader can then have a shorter wait time because it can align well with

the code upload process. However, this design causes the Uno to reset whenever it connects to a new device on Mac OS X or Linux computers via USB. For about half a second after connecting, the bootloader runs, and though it mostly ignores incorrect data, it will capture the first few bytes sent right after connecting. So, if your program needs to receive data immediately after starting, ensure that the software waits for a second before sending any data. To disable the auto-reset feature, you can cut a trace on the Uno, labeled "RESET-EN," and solder the pads to re-enable it. Connecting a 110-ohm resistor between 5V and the reset line is a additional method of stopping the auto-reset; online forums have more information on this.

3.11 USB Overcurrent Protection

A resettable polyfuse on the Arduino Uno protects the USB ports on your computer from short circuits or excessive current. While most computers come with internal safeguards, this fuse provides added protection. If the USB port draws more than 500 milliamps, the fuse disconnects the circuit until the issue is resolved.

3.12 Physical Characteristics

The Arduino Uno's circuit board is 2.7 inches long and 2.1 inches wide, with the power jack and USB connector protruding just a little bit longer. The board can be mounted onto a surface or case using the four screw holes. Notably, digital pins 7 and 8 are separated by 0.16 inches, as opposed to the usual 0.1-inch distance between the other pins.

4. Device Features

4.1 I/O PORTS

The device might have up to five ports available. This number depends on the specific device you use and the features you activate. Some pins in these ports do double duty and are shared with other functions within the device. Typically, when you enable a special feature, you can't use that particular pin for regular tasks.

4.2 Capabilities of I/O Port Pins

When creating an application, it's important to consider what each port pin can do. Some pins can supply more power, which is useful for certain devices. Additionally, some pins can accept voltages higher than the usual power supply level (VDD) as shown in figure 3.

4.3 Power Capacity of Output Pins

Different groups of pins are designed to deliver different power levels to suit various applications. For example, PORTB and PORTC can supply more power, making them suitable for powering devices like LEDs. On the other hand, other ports are aimed at supporting lighter tasks primarily for signal indications. Table 9-1 provides a detailed summary of the output capabilities of each port.

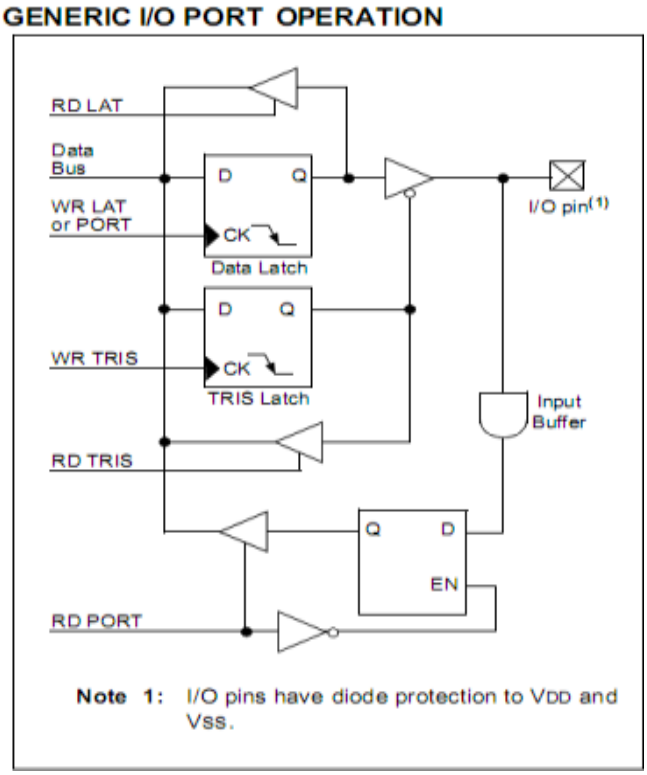


Figure 3. Understanding How I/O Ports Function

4.4 Considerations for Input Pins and Voltage Levels

The purpose of input pins determines their voltage tolerance. As is typical for digital systems, pins used only for digital inputs may withstand voltages of up to 5.5 volts. Pins that double as analog inputs, on the other hand, can only withstand lower voltage levels in order to operate properly which is provided in figure 4.

4.5 PORTA, TRISA, and LATA REGISTERS

The 7-bit port PORTA is utilized for both input and output. It may operate as a 5-bit port, depending on the oscillator mode. By setting the TRISA bit to 1, you can turn a PORTA pin into an input and switch the output driver to high-impedance mode. A pin can be made an output by setting the TRISA bit to 0, which transfers the contents of the output latch to the pin. Writing to the PORTA register updates the port latch

and displays the pins' current state. Data for PORTA is stored in the LATA register, which is mapped in memory. When reading or modifying the LATA register, you work with PORTA's latched output value. Some PORTA pins have multiple functions, acting as both analog inputs and handling other signals like VREF+ and VREF- for voltage reference as delivered in figure 5.

OUTPUT DRIVE LEVELS

Port	Drive	Description
PORTA	Minimum	Intended for indication.
PORTD		
PORTE		
PORTB	High	Suitable for direct LED drive levels.
PORTC		

Figure 4. Output Drive Values

INPUT VOLTAGE LEVELS

Port or Pin	Tolerated Input	Description
PORTA<7:0>	VDD	Only VDD input levels tolerated.
PORTB<3:0>		
PORTC<2:0>		
PORTE<2:0>		
PORTB<7:4>	5.5V	Tolerates input levels above VDD, useful for most standard logic.
PORTC<7:3>		
PORTD<7:0>		

Figure 5. Input Voltage Values

They can also serve as comparator voltage reference outputs. You must modify control bits in the ADCON1 register in order for pins RA0 through RA3 and RA5 to function as analog-to-digital converter inputs. Bits in the CMCON register can be adjusted to use RA0 and RA3 as comparator inputs. Turn off the comparators in order to use RA0 through RA3 as digital inputs. Every PORTA pin has full CMOS output drivers and can handle TTL input levels. Even in analog mode, the TRISA register regulates whether PORTA pins are utilized for input or output. When using the pins as analog inputs, make sure the TRISA register bits stay set.

4.6 PORTB, TRISB, and LATB Registers

Data can be sent and received via the 8-bit port PORTB, and its input/output status is determined by the TRISB register. Set a TRISB bit to 1 to make the pin an input, which means it won't send signals. Set it to 0 to make the pin an output, so it sends signals. The LATB register controls the signals that PORTB sends when acting as an output. If you read, modify, or write to LATB, you change these output signals. PORTB pins have weak pull-up resistors. You can activate them with the RBPU control bit, useful when no signal drives the pins. These resistors aren't active at startup. Some pins (RB<7:4>) can detect changes in input and trigger an interrupt, useful for waking the device when a button is pressed. This interrupt function works with pins set as inputs. A mismatch between signal and saved value triggers the RB Port Change Interrupt. Reading PORTB updates the saved value and clears the interrupt. Use the interrupt feature for waking devices, not for regular signals. Avoid constantly checking PORTB when using the interrupt feature. The RB5 pin can perform different functions like working as a clock input for Timer0 or other special outputs.

4.7 PORTC, TRISC, and LATC Registers

The port PORTC is 8-bit, much like PORTB. The TRISC regulates whether the PORTC pins are outputs (set to 0) or inputs (set to 1). The LATC register determines the output signals of PORTC when pins are outputs. PORTC pins may serve other device functions, and you need to manage settings carefully. These functions may override TRISC settings, determining pin direction without user input. Pins RC4 and RC5 only function as inputs when acting as digital ports. If connected to an external device, they receive signals. Other devices may change settings in TRISC, but you can check current settings by reading TRISC, regardless of other device actions.

4.8 PORTD, TRISD, and LATD Registers

PORTD is another 8-bit port, similar to PORTB and PORTC that can be used for input and output. TRISD register controls direction—setting a TRISD bit to 1 makes the PORTD pin an input, while setting it to 0 makes it an output. When you set a TRISD bit to 0, the corresponding PORTD pin becomes an output. This means that the information stored in the output latch will be shown on that pin. The LATD, or Data Latch register, is linked directly with memory, which allows for reading and writing of the output values for PORTD. Schmitt Trigger input buffers are used by each pin on PORTD, and each pin has a weak internal pull-up that can be turned on for all pins with a single control bit, RDPU (PORTE<7>). If a pin is configured as an output, the weak pull-up automatically turns off, and pull-ups are disabled following a Power-on Reset (POR). These pull-ups can be used for multiple features, like those on PORTB. For the PORTE, TRISE, and LATE registers, their implementation varies with the specific PIC18F46J11 device. PORTE operates as a 3-bit port in devices with 44 pins, and each pin (RE0/AN5/PMRD, RE1/AN6/PMWR, and RE2/AN7/PMCS) can be set as either input or output. When selected as analog inputs, these are read as "0" and also employ Schmitt Trigger input buffers. The

direction of the RE pins is managed by the TRISE register. When a TRISE bit is set to 1, the accompanying PORTE pin becomes an input; when it is set to 0, it becomes an output. Even with analog inputs, you need to ensure pins are set as inputs when necessary. The LATE register works like LATD, facilitating read-modify-write actions that change latched output values for PORTE. The REPU control bit (PORTE<6>) can be used to activate the weak internal pull-up that is present on each PORTE pin. Once a pin becomes an output, the pull- up turns off. Pull-ups deactivate after a POR. The development of microcontrollers traces back to the advances in integrated circuit technology. This allowed numerous transistors to fit onto a single chip, setting the stage for microprocessors. Early computers achieved functionality by adding peripheral elements like memory and input-output lines. As technology advanced, integrated circuits emerged, housing both processors and peripherals. This progress led to the first microcomputer chip, what we call a microcontroller today. Memory in a microcontroller is crucial for data storage, consisting of various locations where data is kept. Addressing is the process of choosing these locations. The tasks involved include selecting a memory location and then accessing its contents. Besides reading, memory allows writing through a control line known as R/W (read/write). When R/W is set to 1, reading occurs; if it's not, writing happens to the location This diagram shows how memory and the CPU work together in a computer system.

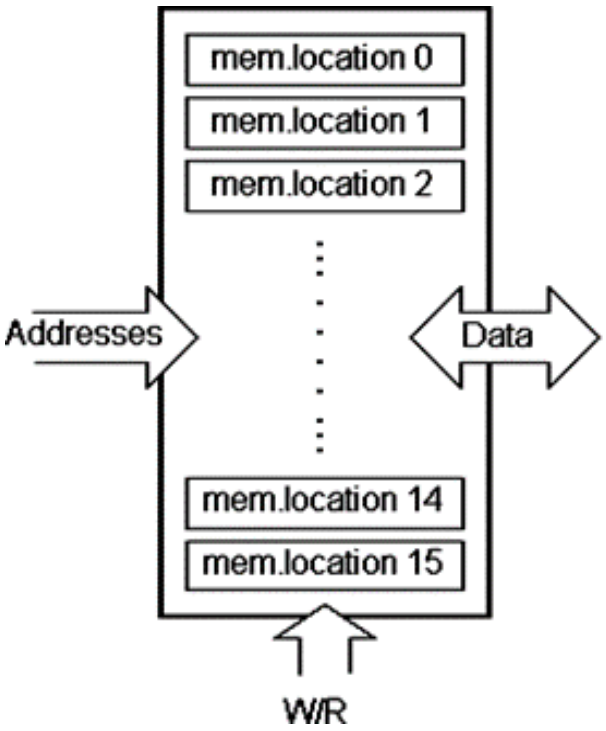
4.9 Central Processing Unit (CPU)

Let's expand a block to include three more memory locations. This block can multiply, divide, subtract, and move data to different memory spots. This part is known as the Central Processing Unit (CPU). In the CPU, these special memory places are called registers [15]. Registers are used to do math and handle other data tasks. Think of a computer having two main parts: the memory and the CPU. They need to share data, but for this to happen smoothly, they use something called a bus. A bus can be eight, sixteen, or more wires that form a roadway.

The address bus and the data bus are the two types of buses. The data bus is as broad as the data, about 8 bits, and the address bus has as many lines as there are memory locations to connect which is displayed in figure 6. The data bus links all of the components inside the microcontroller, whereas the address bus transmits locations from the CPU.

4.10 Input-Output Unit

The new locations we introduced are referred to as ports. Input, output, or both (bidirectional) ports are all possible. You select the port to use before using it. After that, you can send and receive data. Ports act like memory spots where you can write or read data, which can be seen on the microcontroller pins. Just like before, the bus helps the memory and CPU share data as shown in figure 7. The bus is a collection of wires including the address and data buses. These buses ensure that everything in the microcontroller works together and stays connected.



Example of simplified model of a memory unit. For a specific input we get a corresponding output. Line R/W determines wheather we are reading from or writing to memory

Figure 6. Memory Unit Diagram

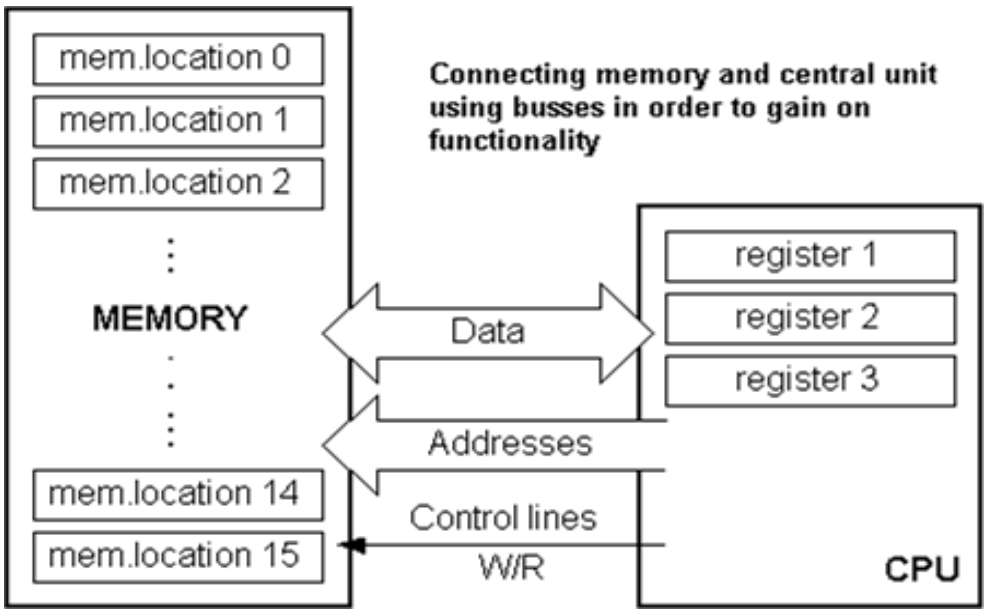


Figure 7. Representation of Bus

4.11 Serial Communication

Priyan and colleagues explain that with different lines for sending and receiving, we can be sent and received data simultaneously. This method is known as full-duplex mode, and it uses something called a serial communication block. In serial communication, data is sent bit by bit, unlike parallel transmission where all data moves at once [16]. When we receive data, we must read it and save it in memory. For sending, the process is the opposite. To ensure this works, we need to use certain rules called protocols. Data is transferred from memory through a path called a bus to send it, and then it goes to the receiver according to these protocols.

4.12 Timer Unit

The timer block helps us understand things like time, duration, and protocol. At its core is something called a free-run counter. Think of this counter as a simple register that counts up by one at regular intervals [17]. By checking the value of this counter at two times, which we'll call T1 and T2, we can figure out how much time has passed between these two moments. Learning how this works is crucial because it's a big part of using the microcontroller, and you'll need to spend some time understanding it. Another important feature is the Watchdog. This ensures the microcontroller runs smoothly and correctly while it's operating which is attached below as figure 8.

In a factory, interference can sometimes cause a microcontroller to stop working properly. Unlike a computer that can be reset with a button, a microcontroller doesn't have an easy reset option. To tackle this, we use a tool called a watchdog. The watchdog acts like a running timer. Our program must write a zero into this timer each time it runs properly. If the program freezes and doesn't write a zero, the timer continues until it reaches its limit. Once it hits this point, it resets the microcontroller, making it start the program over and hopefully work right the next time.

Figure 8. Watch Dog Timer

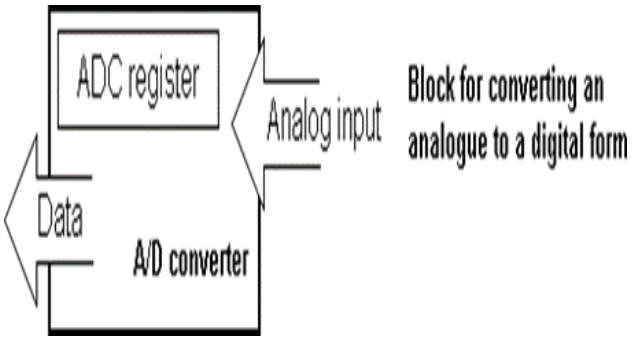


Figure 9. Analog to Digital Converter

4.13. Analog to Digital Converter

Signals from outside sources often aren't in a form that microcontrollers can understand, since microcontrollers read only zeros and ones. We employ an Analog to Digital Converter to transform these

signals into a format that the microcontroller can understand.

An element that converts analog signals, such as voltage, into binary digits (0s and 1s) is called an Analog to Digital Converter (ADC) which is shown in figure 9. These binary numbers are then sent to the CPU for further processing.

4.14 CISC and RISC

The Harvard architecture was developed after the von Neumann architecture to make microcontrollers faster. It keeps the data bus and address bus separate, allowing more data to move quickly through the central processing unit. This setup makes data processing faster because it can handle more at once. Program memory being separate from data memory means instructions are not limited to 8 bits, which adds flexibility [18]. Harvard architecture typically executes faster, frequently in a single cycle, but includes fewer instructions than von Neumann's approach. "RISC microcontrollers" (Reduced Instruction Set Computers) are microcontrollers that use Harvard architecture, whereas "CISC microcontrollers" are those that use von Neumann's design.

4.15 Memory Organization

In PICmicro MCUs, there are three main memory blocks. Due to their separate routes, the Program Memory and Data Memory can be accessed simultaneously without interfering with one another [19].

4.16 Program Memory Organization

The 13-bit program counter used by Arduino UNO devices allows them to operate with an 8K x 14 address program memory area. The FLASH program memory available on the PIC16F877/876 devices is 8K x 14 words, whereas the PIC16F873/874 devices offer 4K x 14 words as shown in the figure 10. It will loop back to the beginning of memory (wraparound) if a memory location that is outside of the available memory is accessed. Address 0000h contains the reset vector, which is used to restart the system, and address 0004h contains the interrupt vector, which manages particular events.

4.17 Data Memory Organization

The various parts of data memory are referred to as banks. General Purpose Registers and Special Function Registers are the two primary register types found in these banks. Two distinct bits, RP1 (from STATUS<6>) and RP0 (from STATUS<5>), are used to choose between these banks. Up to 128 bytes, or 7Fh, can be stored in each bank which is displayed below in figure 11.

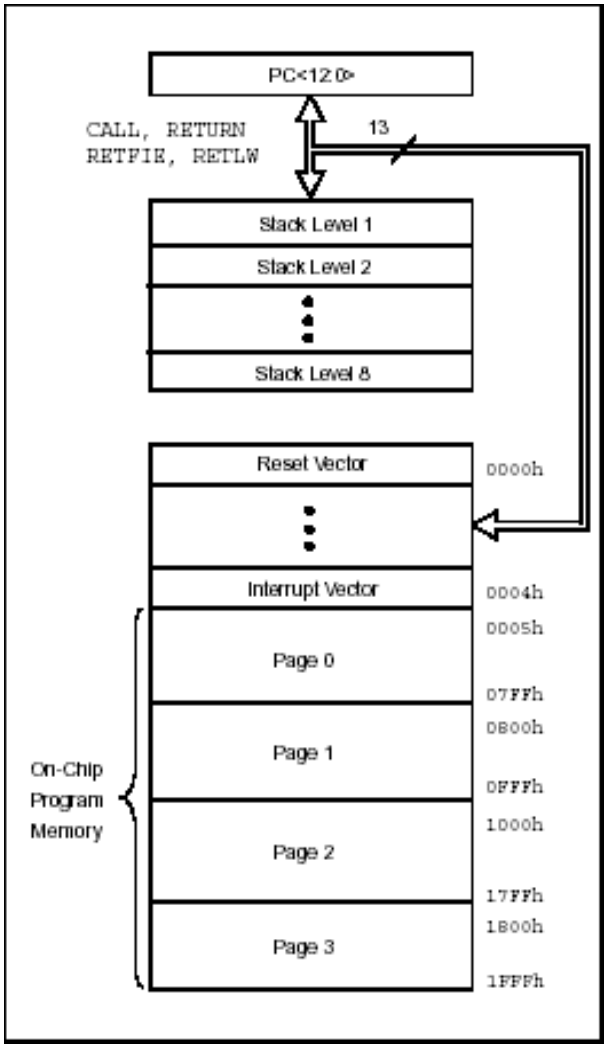


Figure 10. Program Memory and Stack Memory

RP1:RP0	Bank
00	0
01	1
10	2
11	3

Figure 11. Table Register Bank Selection

Special Function Registers are located in the lower portion of each bank. The General Purpose Registers, which serve as static Random Access Memory (RAM), are located above these. The design allows easy access and management of data stored in different banks. This setup ensures that important system functions have their dedicated areas while also allowing flexibility for general data storage needs.

4.18 Special Function Registers

Special Purpose Registers, a form of memory known as static RAM, are utilized by the CPU and its peripheral modules to regulate device activities. These registers are separated into two categories: peripheral registers and core (CPU) registers.

4.19 Status Register

The data memory bank selection bits, reset status, and the outcomes of arithmetic operations are all stored in the STATUS register. Like other registers, it can be the target of instructions. However, if an instruction impacts the Z, DC, or C bits, writing back to the STATUS register won't change these bits, since updating them is blocked. They change only through device logic. The TO and PD bits can't be altered, so using the STATUS register might sometimes yield unexpected outcomes.

4.20 I²C Protocol

Inter-Integrated Circuit, or I²C for short, was created by Philips to connect slower devices, like as sensors, to a cell phone, motherboard, or embedded system. It requires just two lines: one for data (SDA) and another for the clock (SCL), with resistors that pull them up to the voltage level. Utilizing a 7-bit address system, I²C can accommodate up to 112 devices on the bus, though address limitations and a bus capacitance ceiling of 400 pF may restrict this number. Designed for simplicity and efficiency, the system facilitates communication among various chip types. All devices on the bus are assigned unique addresses, enabling a master-slave communication model. The I²C bus supports multiple masters and includes collision detection to prevent errors during simultaneous data transfers. 3.4 Mbit/s in High-Speed mode, 400 kbit/s in Fast mode, and 100 kbit/s in Standard mode are the maximum data transmission speeds possible on the I²C bus. The system also features filtering to eliminate noise and maintain data accuracy. The limit for connected devices is linked to a maximum bus capacitance of 400 pF. In systems that use 8-bit microcontrollers for digital control, a serial bus structure is preferred because it needs less wiring compared to parallel buses. Although serial buses transmit data more slowly, they simplify connections and wiring. A bus is not just an interconnection of wires but includes all the communication rules within a system. A clear protocol is vital to avoid data confusion or loss. Faster and slower devices must communicate without issues, and the system should allow easy updates without altering current connections. There must also be a way to identify which gadget is in charge of the bus and when. The source of the bus clock needs to be specified if devices operate at various rates. The I²C bus specification takes all of these factors into account. With two active wires and a ground line, the I²C bus is a straightforward configuration. SDA and SCL are the names of these two active lines. Serial data lines are referred to as SDAs, and serial clock lines are referred to as SCLs. These cables are bi-directional, meaning they can transmit and receive signals. Each device that is connected to the bus, be it an ASIC, memory chip, LCD driver, or microcontroller (MCU), has a distinct address. These devices can send data, receive data, or do both, based on what they are designed to do. For example, an LCD driver is only made to receive data, while a memory chip can both send and receive data. One special feature of the I²C bus is that it allows more than one device to start a data transfer. This is why it's called a multi-master bus. The device that starts the transfer is known as the Bus Master. All the other devices involved in the data transfer at that moment are called Bus Slaves.I²C- compatible devices make it easy to design systems. They let you quickly go from a plan to a working model. These devices can be added or removed from the I²C bus directly, without needing any extra connections. This makes it simple to upgrade or change the system by clipping devices on or off the bus. Some important features of I²C-compatible devices are:

- They work with any chip-making technology, such as NMOS, CMOS, or bipolar processes.
- Information is carried over the SDA and SCL wires.
- Every device, be it a memory chip, LCD driver, microcontroller, or keyboard interface, has a unique address that makes it simple to identify.

As shown in figure 12 I²C-bus is a multi-master bus since it can be controlled by multiple devices. So, you can connect several devices that can manage the bus at once. This might lead to situations where multiple microcontrollers try to start sending data at the same time. To prevent problems in such cases, an arbitration process is in place. All I²C interfaces are connected to the bus via a wired-AND connection in this operation. The master who tries to send a "one" when another sends a "zero" will lose if several masters try to put data on the bus at the same time. The wired-AND connection to the SCL line is used to aggregate the clock signals from the masters during arbitration. When data is being transferred, clock signals are produced by masters. If another master takes part in arbitration or if a slow-slave device holds the clock line down, these signals may alter.

Because the SDA and SCL lines are bi-directional, signals can be sent and received. As seen in Figure 13, they are connected to a positive supply voltage via a current source or pull-up resistor. Both lines are HIGH when the bus is not moving. The wired-AND function requires an open-drain or open-collector for device outputs on the bus. Standard mode on the

I²C-bus can transfer data at up to 100 kbit/s, fast mode at up to 400 kbit/s, and high-speed mode at up to 3.4 Mbit/s.

The MSSP module in I²C mode is very useful. It handles master and slave tasks. It supports general calls and uses interrupts to notify you when Start and Stop signals happen. This is helpful to see if the bus is free, especially when there are multiple masters. It works with the standard mode specifications and can address using both 7-bit and 10-bit formats.

4.21 Display Unit

The cheapest option for a display is an LCD. However, the Raspberry Pi offers more flexibility since you can connect various display units directly through its display port.

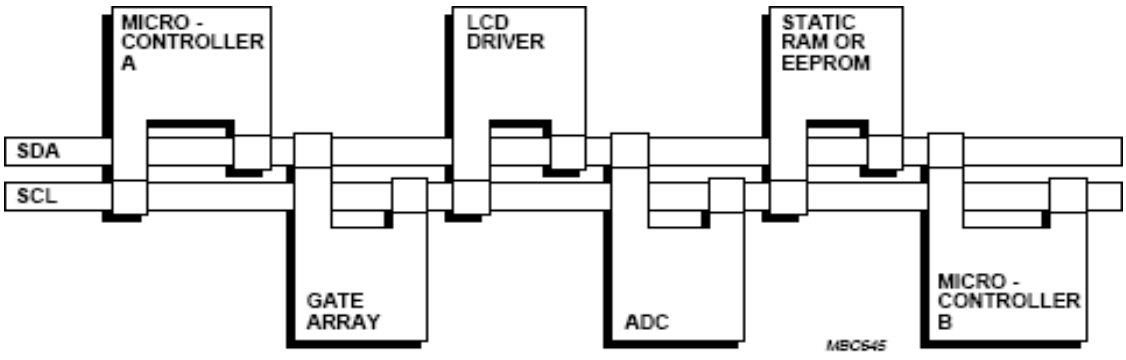
4.22 Power Supply

This power adapter is designed to deliver 12 Volts at 1 Amp. It takes 100-240 Volts AC as input and converts it to 12 Volts DC. It's ideal for many devices like CCTV cameras and wireless routers. Built-in protections prevent short circuits, over-voltage, and over-current issues. Made for Indian power sockets, it eliminates the need for a plug converter. It is small, light, and reliable, providing a stable voltage. Its efficiency is high, keeping energy use low. Excellent for gadgets like LED strips, routers, modems, and mobile phones. If you're replacing an old adapter, ensure the DC socket size and power rating match your device.

Definition of I²C-bus terminology

TERM	DESCRIPTION
Transmitter	The device which sends data to the bus
Receiver	The device which receives data from the bus
Master	The device which initiates a transfer, generates clock signals and terminates a transfer
Slave	The device addressed by a master
Multi-master	More than one master can attempt to control the bus at the same time without corrupting the message
Arbitration	Procedure to ensure that, if more than one master simultaneously tries to control the bus, only one is allowed to do so and the winning message is not corrupted
Synchronization	Procedure to synchronize the clock signals of two or more devices

Figure 12. I²C-Bus Terms



Example of an I²C-bus configuration using two microcontrollers.

Figure 13. I²C Bus Configuration

4.23 Applications

This adapter provides up to 12 Volts and 1 Amp, making it a smart choice for LEDs, routers, and portable players. Its lower power outputs like 12v 0.5A can also be replaced. Check if the power requirements are compatible with your gadget, as this adapter works with specific current outputs. Perfect for routers, including several Netgear models. Double-check with your router's supplier to confirm compatibility, as we can't ensure it fits every model available.

4.24 Jump Wire

A jump wire is used to link the test board, prototype, or internal circuits to other devices that aren't connected yet.

4.25 MEMS Sensor

We are using the MEMS Sensor MMA7600FC to detect falls, which can happen due to changes in blood pressure. This sensor helps us know if a fall has occurred. MEMS stands for Micro Electro Mechanical Systems, involving tiny devices with moving parts. These systems are very small and connect to nanotechnology advancements. In Japan, MEMS are called micro machines, while in Europe, they are known as micro systems technology (MST). Components in MEMS range from 1 to 100 micrometers, and the devices themselves can be between 20 micrometers and 1 millimeter. These systems have a central unit, known as a microprocessor, and several micro sensors that interact with the environment. Due to their tiny size, design considerations like electrostatic charges and fluid dynamics, such as surface tension and viscosity, become quite important compared to larger mechanical devices.

4.26 Heartbeat Sensor

The heartbeat sensor is designed to provide a digital reading of the heartbeat when you place a finger inside it. This digital signal can be directly linked to an Arduino board to calculate the Beats per Minute (BPM). The sensor works on the principle of changes in light intensity due to blood flow in the finger with each heartbeat. The sensor employs an IC known as LM358, which includes a very bright red LED and a light detector. One part functions as an amplifier, and the other serves as a comparator. The LED needs extra brightness to penetrate the finger, allowing the light detector on the opposite side to catch it. Less light reaches the detector because the finger becomes somewhat less transparent while the heart beats and blood flows. This variation alters the detector's signal, which is then transformed into an electrical pulse.

4.27 Temperature Sensor

The output voltage of the LM35 series precision temperature sensors is directly proportional to the temperature in Celsius. Unlike sensors that use the Kelvin scale, the LM35 has the advantage of providing Celsius values without requiring adjustment for a large constant voltage. This sensor offers accurate readings without the need for extra calibration or trimming. According to the LM35 datasheet, these sensors are accurate, with ±1/4°C accuracy at room temperature and ±3/4°C accuracy throughout a broad temperature range of -55°C to +150°C.

4.28 Vibration Motor (DC, 12V)

For this study, we need a water pump that operates on 12V DC power. DC motors are commonly used due to their compatibility with direct current power systems. In these motors, some parts hold magnets while the stationary part carries conductors. Supports inside the motor allow it to keep turning continuously around its center axis. The motor’s design incorporates components that facilitate smooth operation and are essential for continuous movement.

5. Software Description

5.1 Use of Open Source Technology

One of the key reasons Linux is widely adopted is because it's open source. Open source means anyone can access, use, and modify the program's source code without paying. This collaborative approach allows people to improve the code and share enhancements within a community. Linux's kernel is under the GNU General Public License (GPL), which emphasizes that "free software" means freedom, not price. This means the Linux kernel can be downloaded and installed at no cost. Being open source also makes the software reliable, flexible, and supported by a big community. Importantly, it's free, helping to lower development costs significantly.

5.2 Operating System

An operating system (OS) is software that allows other programs to run on a computer. It plays a crucial role in managing software and hardware resources, acting like the computer's backbone. The OS handles everything from allocating memory and processing inputs from devices to displaying outputs. It also manages file storage on hard drives and controls peripherals like cameras and printers. By doing all this, the OS ensures the computer runs smoothly even when multiple programs or devices are in use simultaneously. Moreover, the OS plays a vital role in security by blocking unauthorized access to the computer system.

5.3 HTML

Hypertext Mark-up Language (HTML) is essential for creating web documents. It involves using tags to describe different parts of the document. These elements give web browsers instructions on how to present images and text on webpages. In order to construct and develop websites, HTML is essential [6].

5.4 MPLAB IDE

An all-in-one tool for editing, project management, and designing for ARDUINO MCU and DSC embedded systems is the MPLAB Integrated Development Environment (IDE). This Windows-based program is used to create applications for digital signal controllers and microchip microcontrollers. It provides a unified platform for embedded system code development and management as an IDE. In MPLAB IDE, programmers use Embedded C language. This language is an extension of standard C, tailored for embedded system needs. It often includes nonstandard features to support specific tasks like fixed-point arithmetic and basic input/output operations. Code speed depends on the processor's power and time constraints, while code size is limited by memory and language use. Embedded software is crucial for each processor, acting like the brain of embedded systems. If the hardware is the body, then the processor is the brain, and the software is the soul. The primary goal of embedded software is to offer the most functionality while consuming the least amount of time and space. During manufacturing, it includes all required device drivers, tailored for specific hardware [7].

5.5 C18 Compiler

An ANSI C-standard tool for Arduino microcontrollers is the MPLAB C18 compiler. When needed, it somewhat deviates from standard ANSI in order to optimize for ARDUINO support. The C18 compiler integrates easily with the MPLAB IDE and operates as a 32-bit Windows application. With the aid of programs like the MPLAB SIM simulator, MPLAB ICD 2 in-circuit debugger, and MPLAB ICE in-circuit emulator, it permits safe debugging.

5.6 MPLAB C18 includes these features

- It follows ANSI '89 standards for coding.
- You can easily integrate it with the MPLAB IDE. This makes managing projects and debugging at the source level straightforward.
- It lets you generate re-locatable object modules. This is great for reusing code more effectively.
- It works well with object modules created by the MPASM assembler. This compatibility means you can mix assembly language and C programming in one project without any trouble.
- You have easy read and write access to external memory.
- It has strong support for using inline assembly, which is important when you need total control over your code.
- The efficient code generator engine optimizes code at several levels, making it more efficient.
- There are many libraries available, including ones for PWM, SPI™, I²C™, UART, USART, string handling, and math functions.

6. Experimental Result

This study aimed at creating and testing a health monitoring system. The system uses a microcontroller and several sensors as shown in the block diagram. It monitors heart rate, body movement, and temperature. Sensors collect this data and send it to the microcontroller. The information is shown on an LCD screen, allowing you to see it live.

A motor driver is included to manage a vibration motor that gently shakes or buzzes if health readings become too high or unsafe. To set up the system, we connected a heartbeat sensor, a MEMS sensor for movement detection, and a temperature sensor to the microcontroller. Each component was checked for proper function. The LCD display shows live health information, while the motor driver manages the vibration alerts if there are issues.

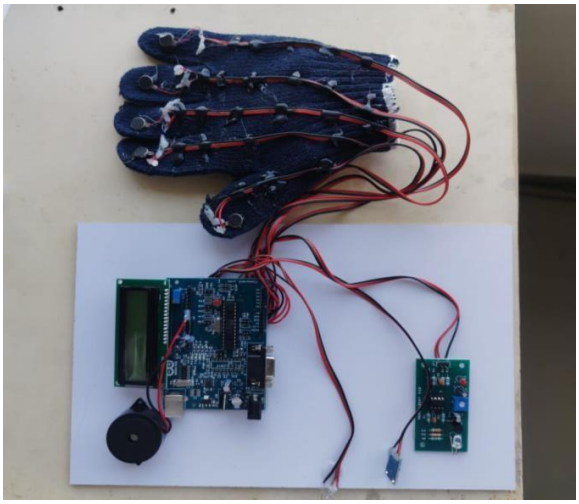


Figure 14. Experimental kit

Testing showed that the system is accurate, efficient, and reacts promptly to health data changes. The microcontroller's fast processing helps spot and show any problems quickly. The vibration alert is particularly useful for those needing immediate

awareness of health issues, such as older adults or those with medical conditions.

Heartbeat Sensor Testing: We tested the heartbeat sensor on several people and checked its readings against a standard pulse oximeter. The results were mostly the same, showing only small differences that were within safe limits.

MEMS Sensor Evaluation: We evaluated the motion sensor by mimicking various movements, including quick falls. The system was able to notice these motion changes and accurately sent information to the microcontroller.

Temperature Sensor Validation: We verified the temperature sensor by comparing its readings with a clinical thermometer. The sensor proved to be very accurate, showing only tiny differences of about ±0.5°C, which confirmed it works well which is displayed in figure 15.

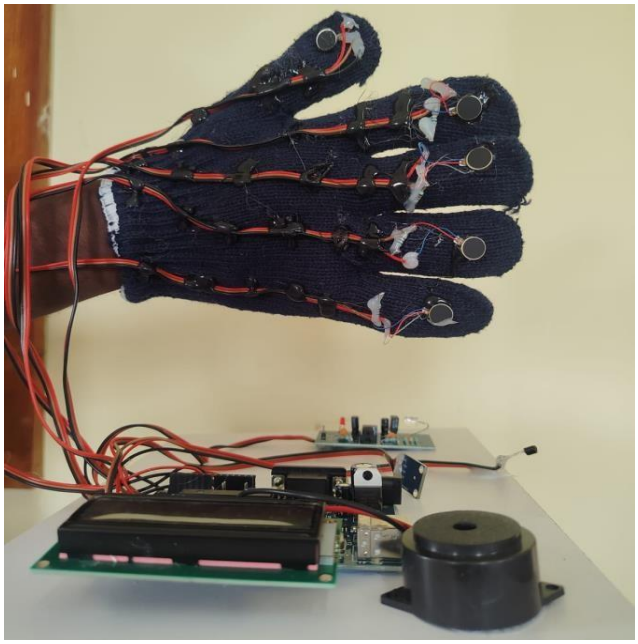


Figure 15. Working pronation position

As shown in 16 this health monitoring system reliably tracks heart rate, movement, and temperature. It's helpful for watching over patients from afar, keeping up with fitness activities, and providing emergency alerts. In the future, it might be even better with features like wireless data transfer and a connection to a mobile app, making it much easier to use and access the information you need.

LCD Display Performance: The display showed real-time updates effectively. This ensured that health information was clear and easy to read at any moment as shown in figure 17.

Vibration Motor Response: The motor was set up to activate when serious conditions were found. It gave instant feedback you could feel when the heart rate, temperature, or motion readings went beyond the set limits, making sure alerts were given in time.

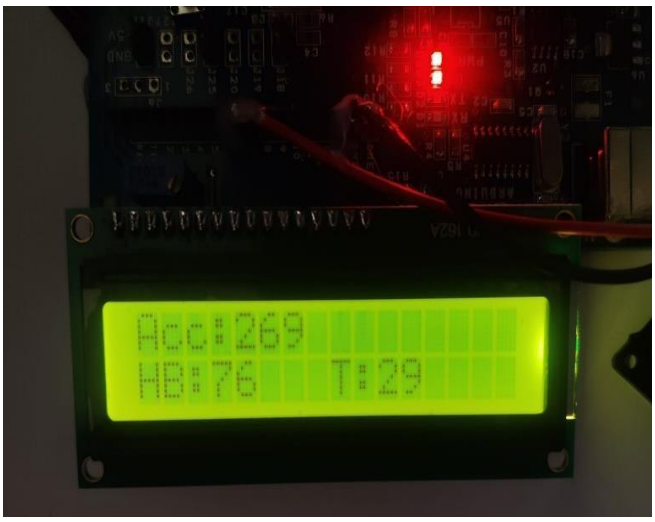


Figure 16. Pronation Position Values

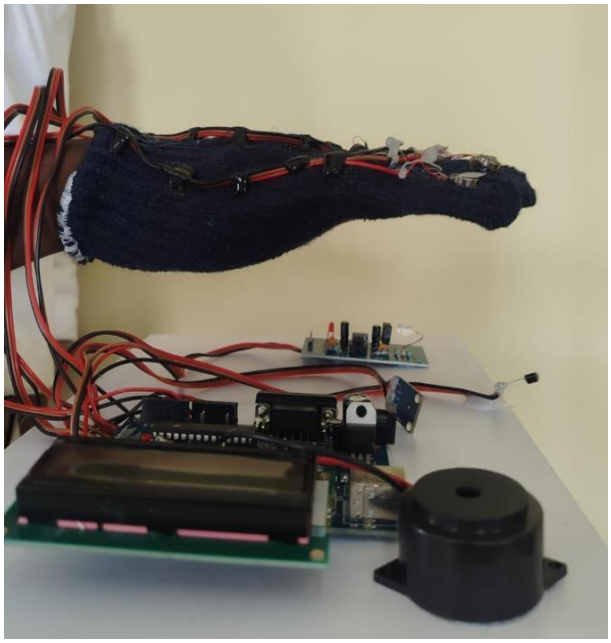


Figure 17. Not working neutral position

7. Conclusion and Future Scope

7.1 Conclusion

We designed and tested gloves specifically for stroke survivors and those with physical disabilities. They fit any hand size, allowing anyone to use them. The gloves help speed up recovery following a stroke. Initial tests have shown positive results. We also compared how quickly recovery happens with our gloves versus existing hospital systems.

7.2 Future Scope

In the future, we aim to enhance the gloves with additional sensors. These could measure temperature, breathing, and pulse, offering continuous health monitoring. Adding a GPS feature would allow real-time location tracking. By replacing the Bluetooth module with IoT technology, patient monitoring could happen from any location.

7.3 Applications

- Children with Special Needs: The model can monitor health and location for children with special needs.
- Elderly People: It can track the health and location of senior citizens.
- Remote and Rural Populations: The system benefits people in areas with limited healthcare access, offering needed monitoring in these regions.

References

[1] G.R.S. Murthy, R.S. Jadon, A review of vision based hand gestures recognition. International Journal of Information Technology and Knowledge Management, 2(2), (2009) 405-410.

[2] P. Garg, N. Aggarwal, S. Sofat. Vision based hand gesture recognition. World academy of science, engineering and technology, 49(1), (2009) 972-977.

[3] F. Karray, M. Alemzadeh, J. Abou Saleh, M.N. Arab, Human-computer interaction: Overview on state of the art. International journal on smart sensing and intelligent systems, 1(1), (2008) 137-159. [DOI]

[4] M.M. Hasan, P.K. Misra, Brightness Factor Matching For Gesture Recognition System Using Scaled Normalization. International Journal of Computer Science & Information Technology (IJCSIT), 3(2), (2019) 99-167. [DOI]

[5] X. Li. (2019) Gesture Recognition Based on Fuzzy C- Means Clustering Algorithm, Master's Thesis or Dissertation, Department of Computer Science. The University of Tennessee Knoxville.

[6] S. Mitra, T. Acharya. Gesture Recognition: A Survey. IEEE Transactions on systems, Man and Cybernetics, Part C: Applications and reviews, 37(3), (2017) 311- 324. [DOI]

[7] S.G. Wysoski, M.V. Lamar, S. Kuroyanagi, A. Iwata, (2019) A Rotation Invariant Approach On Static- Gesture Recognition Using Boundary Histograms And Neural. IEEE Proceedings of the 9th International Conference on Neural Information Processing, Singapura. [DOI]

[8] Joseph J. LaViola, (1999) A Survey of Hand Posture and Gesture Recognition Techniques and Technology. Brown University, United States.

[9] R.Z. Khan, N.A. Ibraheem, (2012) Survey on gesture recognition for hand image postures. Computer and information science, 5(3), 110. [DOI]

[10] T.B. Moeslund, E. Granum, A Survey of Computer Vision-Based Human Motion Capture. Computer Vision and Image Understanding, 81, (2019) 231– 268. [DOI]

[11] N.A. Ibraheem, R.Z. Khan, M.M. Hasan, Comparative study of skin color based segmentation techniques. International Journal of Applied Information Systems (IJAIS), 5(10), (2013) 24-38.

[12] M. Elmezain, A. Al-Hamadi, J. Appenrodt, B. Michaelis, A hidden markov model-based isolated and meaningful hand gesture recognition. International Journal of Electrical, Computer, and Systems Engineering, 3(3), (2009) 156-163.

[13] E. Stergiopoulou, N. Papamarkos, Hand gesture recognition using a neural network shape fitting technique. Engineering Applications of Artificial Intelligence, 22(8), (2009) 1141-1158. [DOI]

[14] M.M. Hasan, P.K. Mishra, HSV brightness factor matching for gesture recognition system. International Journal of Image Processing (IJIP), 4(5), (2010) 456-467.

[15] Malima, Ozgur, Cetin. (2006). A fast algorithm for vision-based hand gesture recognition for robot control. IEEE 14th Signal Processing and Communications Applications, IEEE, Turkey. [DOI]

[16] M.M. Hasan, P.K. Mishra, Features fitting using multivariate gaussian distribution for hand gesture recognition. International Journal of Computer Science & Emerging Technologies IJCSET, 3(2), (2012) 73-80.

[17] M.M. Hasan, P.K. Mishra, Robust gesture recognition using gaussian distribution for features fitting. International Journal of Machine Learning and Computing, 2(3), (2012) 266.

[18] W.T. Freeman, M. Roth, Orientation histograms for hand gesture recognition. In International workshop on automatic face and gesture recognition, 12, (1995) 296-301.

[19] B.W. Min, H.S. Yoon, J. Soh, Y.M. Yang, T. Ejima, (1997) Hand gesture recognition using hidden Markov models. In 1997 IEEE International Conference on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation, IEEE, USA. [DOI]

Funding

No funding was received for the conduct of this research

Conflict of interest

The authors declare that no conflict of interest exists.

Does the Article Screened for Similarity?

Yes.

About the License