Advanced and Intelligent Wheel chair control system

Ms. S Manjula, Shaik Juveriya Tasneem, Nellikondi Sravani, Ellavula Pravalika

[1]Associate Professor,ECE Department Bhoj Reddy Engineering College for Women

[2,3,4]B. Tech Students, Department Of ECE, Bhoj Reddy Engineering College For Women, India.

manjula.sathineni@slv-edu.in

ABSTRACT

This project aims to design a prototype wheelchair that offers multiple control methods for enhanced accessibility and ease of use. The wheelchair can be controlled via a joystick and four pushbuttons for manual operation, while also incorporating MEMS sensors for wireless control through RF (Radio Frequency) technology. To ensure safety, the wheelchair is equipped with an IR sensor that detects obstacles in its path and triggers an alert using a buzzer. The movement of the wheelchair can be directed in multiple ways, including forward, reverse, left, right, and stop, providing the user with flexible and responsive navigation options. This design aims to improve the mobility and independence of individuals with physical disabilities, offering both manual and remote-control functionalities for ease of use in various environments.

1-INTRODUCTION

The purpose of this project is to control a wheel chair using joystick, MEMS Sensor, Push Buttons and detecting obstacles using IR obstacle Sensor. The obstacle detection mechanism is done by an IR obstacle sensor that makes use of IR rays to find the presence of an obstacle in its path and activate the Buzzer for alerts.

To control the wheel chair by using joystick and detecting obstacles using audio signaling device. Joystick is a simple device with four direction movement. The wheelchair is equipped with a joystick and four pushbuttons for manual control, along with a MEMS (Micro-Electro- Mechanical Systems) sensor for wireless control using RF (Radio Frequency) technology. Another method of controlling using the Head movement through the MEMS sensor.

To ensure safety, the wheelchair incorporates an IR sensor to detect obstacles and alert the user via buzzer. The movement of the wheelchair can be controlled in various directions, including forward, reverse, left, right, and stop, providing flexible and responsive navigation. It can be made to produce an analog voltage which is processed by the microcontroller to produce the corresponding digital output with the help of inbuilt ADC. Microcontroller will continuously read the data from joystick. Based on the received command, Microcontroller will control the direction (like left, right, front, back and stop) of the wheel chair through DC motors. To ensure safety, the wheelchair is equipped with an IR sensor that detects obstacles in its path and triggers an alert using a Buzzer. The controlling device of the whole unit is a Microcontroller to which input and output modules are interfaced. The Microcontroller is programmed in Embedded C language which intelligently performs the specific task. Here, the Microcontroller gets input from the IR obstacle sensor attached to the wheel chair. This input is processed by controller and acts appropriately on the motors of the wheel chair. Additionally, a MEMS sensor is integrated into the system to enable wireless or motion-based control, providing an advanced and flexible method for users with limited

hand mobility. This combination of input methods enhances the overall functionality and user-friendliness of the intelligent wheelchair system.

2-EMBEDDED SYSTEMS

Embedded Systems:

An embedded system is a computer system designed to perform one or a few dedicated functions often with real-time computing constraints. It is embedded as part of a complete device often including hardware and mechanical parts. By contrast, a general-purpose computer, such as a personal computer (PC), is designed to be flexible and to meet a wide range of end-user needs. Embedded systems control many devices in common use today.

Embedded systems are controlled by one or more main processing cores that are typically either microcontrollers or digital signal processors (DSP). The key characteristic, however, is being dedicated to handle a particular task, which may require very powerful processors. For example, air traffic control systems may usefully be viewed as embedded, even though they involve mainframe computers and dedicated regional and national networks between airports and radar sites. (Each radar probably includes one or more embedded systems of its own.)

Since the embedded system is dedicated to specific tasks, design engineers can optimize it to reduce the size and cost of the product and increase the reliability and performance. Some embedded systems are mass-produced, benefiting from economies of scale.

Physically embedded systems range from portable devices such as digital watches and MP3 players, to large stationary installations like traffic lights, factory controllers, or the systems controlling nuclear power plants. Complexity varies from low, with a single microcontroller chip, to very high with multiple units, peripherals and networks mounted inside a large chassis or enclosure.

In general, "embedded system" is not a strictly definable term, as most systems have some element of extensibility or programmability. For example, handheld computers share some elements with embedded systems such as the operating systems and microprocessors which power them, but they allow different applications to be loaded and peripherals to be connected. Moreover, even systems which don't expose programmability as a primary feature generally need to support software updates. On a continuum from "general purpose" to "embedded", large application systems will have subcomponents at most points even if the system as a whole is "designed to perform one or a few dedicated functions", and is thus appropriate to call "embedded". A modern example of embedded system is shown in fig: 2.1.

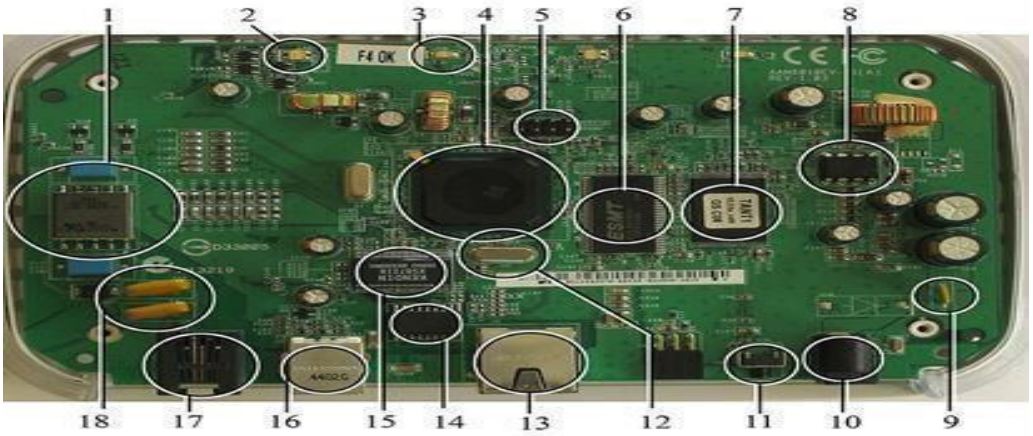

Fig 2.1: A modern example of embedded system

Labeled parts include 1. Power LED 2. Status LED 3.User Interface 4.Microphone 5.Speaker. 6.External Connector 7.Antenna 8.Camera 9.Flash Memory 10.Microcontroller 11.Processor 12.Memory (RAM)13.Sensors 14.Actuators 15.Battery 16.Power Management IC 17.Network Interface 18.Expansion Connector. The hardware for the system is usually chosen to make the device as cheap as possible. Spending an extra dollar a unit in order to make things easier to program can cost millions. Hiring a programmer for an extra month is cheap in comparison. This means the programmer must make do with slow processors and low memory, while at the same time battling a need for efficiency not seen in most PC applications. Below is a list of issues specific to the embedded field.

Tools:

Embedded development makes up a small fraction of total programming. There's also a large number of embedded architectures, unlike the PC world where 1 instruction set rules, and the UNIX world where there's only 3 or 4 major ones. It also means that they're lowering featured, and less developed. On a major embedded project, at some point you will almost always find a compiler bug of some sort. In embedded system development, tools play a crucial role but differ significantly from those used in traditional software development. Unlike mainstream software environments, embedded development makes up only a small fraction of the total programming world. As a result, there is a wide variety of embedded architectures and hardware platforms.

Need for Embedded Systems:

The uses of embedded systems are virtually limitless, because every day new products are introduced to the market that utilizes embedded computers in novel ways. In recent years, hardware such as microprocessors, microcontrollers, and FPGA chips have become much cheaper. So when implementing a new form of control, it's wiser to just buy the generic chip and write your own custom software for it. Producing a custom-made chip to handle a particular task or set of tasks costs far more time and money. Many embedded computers even come with extensive libraries, so that "writing your own software" becomes a very trivial task indeed. From an implementation viewpoint, there is a major difference between a computer and an embedded system. Embedded systems are often required to provide Real-Time response. The main elements that make embedded systems unique are its reliability and ease in debugging.

Debugging:

Embedded debugging may be performed at different levels, depending on the facilities available. From simplest to most sophisticate, they can be roughly grouped into the following areas:

- Interactive resident debugging, using the simple shell provided by the embedded operating system (e.g. Forth and Basic)
- External debugging using logging or serial port output to trace operation using either a monitor in flash or using a debug server like the Remedy Debugger which even works for heterogeneous multi core systems.
- An in-circuit debugger (ICD), a hardware device that connects to the microprocessor via a JTAG or Nexus interface. This allows the operation of the microprocessor to be controlled externally, but is typically restricted to specific debugging capabilities in the processor.
- An in-circuit emulator replaces the microprocessor with a simulated equivalent, providing full control over all aspects of the microprocessor.
- A complete emulator provides a simulation of all aspects of the hardware, allowing all of it to be

controlled and modified and allowing debugging on a normal PC.

- Unless restricted to external debugging, the programmer can typically load and run software through the tools, view the code running in the processor, and start or stop its operation. The view of the code may be as assembly code or source-code.

Because an embedded system is often composed of a wide variety of elements, the debugging strategy may vary. For instance, debugging a software (and microprocessor) centric embedded system is different from debugging an embedded system where most of the processing is performed by peripherals (DSP, FPGA, co-processor). An increasing number of embedded systems today use more than one single processor core. A common problem with multi-core development is the proper synchronization of software execution.

3-HARDWARE DESCRIPTION

In this chapter the block diagram of the project and design aspect of independent modules are considered. Block diagram is shown in fig: 3.1.

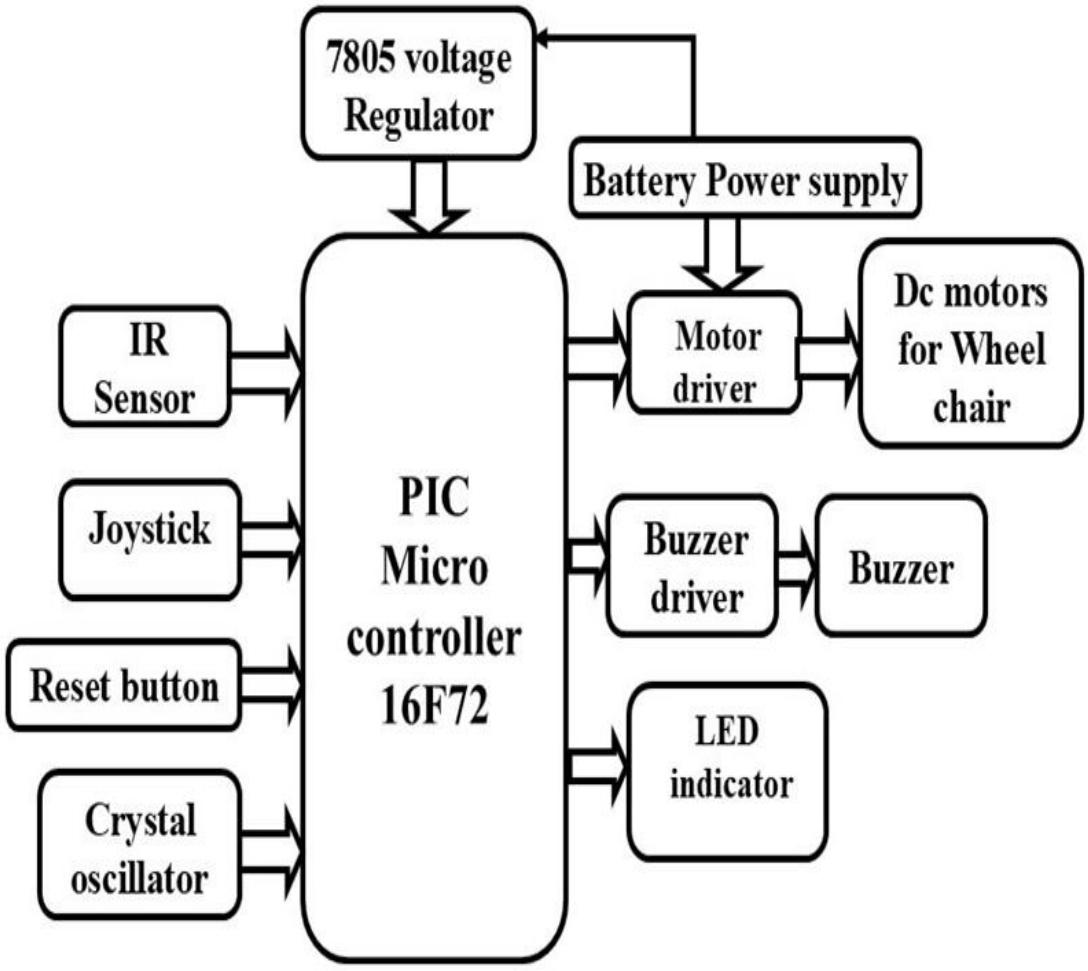

Fig 3.1: Block diagram (Receiver) of Advanced and Intelligent wheel chair control system

BLOCK DIAGRAM OF THE PROJECT

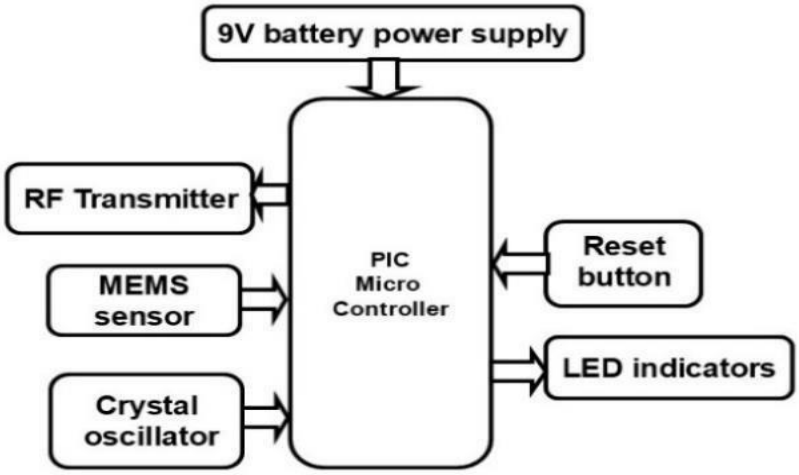

Fig 3.1.2: Block diagram (Transmitter) of Advanced and Intelligent wheel chair control system using MEMS Sensor

4-SOFTWARE DESCRIPTION

This project is implemented using following software's:

- Express PCB – for designing circuit
- PIC C compiler - for compilation part

Express PCB:

Breadboards are great for prototyping equipment as it allows great flexibility to modify a design when needed; However the final product of a project, ideally should have a neat PCB, few cables, and survive a shake test. Not only is a proper PCB neater but it is also more durable as there are no cables which can yank loose. Express PCB is a software tool to design PCBs specifically for manufacture by the company Express PCB (no other PCB maker accepts Express PCB files). It is very easy to use, but it does have several limitations.

Preparing Express PCB for First Use:

Express PCB comes with a less then exciting list of parts. So before any project is started head over to Audio logical and grab the additional parts by morsel, ppl, and tangent, and extract them into your Express PCB directory. At this point start the program and get ready to setup the workspace to suit your style. Click View -> Options. In this menu, setup the units for "mm" or "in" depending on how you think, and click "see through the top copper layer" at the bottom. The standard color scheme of red and green is generally used but it is not as pleasing as red and blue.

The Interface:

When a project is first started you will be greeted with a yellow outline. This yellow outline is the dimension of the PCB. Typically after positioning of parts and traces, move them to their final position and then crop the PCB to the correct size. However, in designing a board with a certain size constraint, crop the PCB to the correct size before starting.

PIC Compiler:

PIC compiler is software used where the machine language code is written and compiled. After compilation, the machine source code is converted into hex code which is to be dumped into the microcontroller for further processing. PIC compiler also supports C language code.

It's important that you know C language for microcontroller which is commonly known as Embedded C. As we are going to use PIC Compiler,

hence we also call it PIC C. The PCB, PCM, and PCH are separate compilers. PCB is for 12-bit opcodes, PCM is for 14-bitopcodes, and PCH is for 16-bit opcode PIC microcontrollers. Due to many similarities, all three compilers are covered in this reference manual. Features and limitations that apply to only specific microcontrollers are indicated within. These compilers are specifically designed to meet the unique needs of the PIC microcontroller. This allows developers to quickly design applications software in a more readable, high-level language. When compared to a more traditional C compiler, PCB, PCM, and PCH have some limitations. As an example of the limitations, function recursion is not allowed.

This is due to the fact that the PIC has no stack to push variables onto, and also because of the way the compilers optimize the code. The compilers can efficiently implement normal C constructs, input/output operations, and bit twiddling operations. All normal C data types are supported along with pointers to constant arrays, fixed point decimal, and arrays of bits.

PIC C is not much different from a normal C program. If you know assembly, writing a C program is not a crisis. In PIC, we will have a main function, in which all your application specific work will be defined. In case of embedded C, you do not have any operating system running in there. So you have to make sure that your program or main file should never exit. This can be done with the help of simple while (1) or for (;;) loop as they are going to run infinitely. We have to add header file for controller you are using, otherwise you will not be able to access registers related to peripherals.

5-RESULT AND DISCUSSION

Output

This project outlines the design of a multi-model wheelchair that prioritizes accessibility and ease of use for individuals with disabilities.

Multiple Control Methods:

- Manual Control:
  - ➢ Joystick for intuitive navigation.
  - ➢ Four pushbuttons for directional commands (forward, reverse, left, right, stop).
  - ➢ MEMS Sensor for directional commands by using head movement.
- Obstacle Detection:
  - ➢ IR sensor for detecting obstacles.
  - ➢ Buzzer to alert the user of impending obstacles.
- Movement Capabilities:
  - ➢ Forward, reverse, left, right, and stop movements for flexible navigation.

Overall, this project aims to develop a wheelchair that significantly improves the quality of life for individuals with disabilities by providing them with increased mobility, independence, and safety. This is a well-defined project with a strong focus on user needs and technological innovation.

Fig 1: Advanced and Intelligent Wheel Chair equipped with joystick and IR sensor

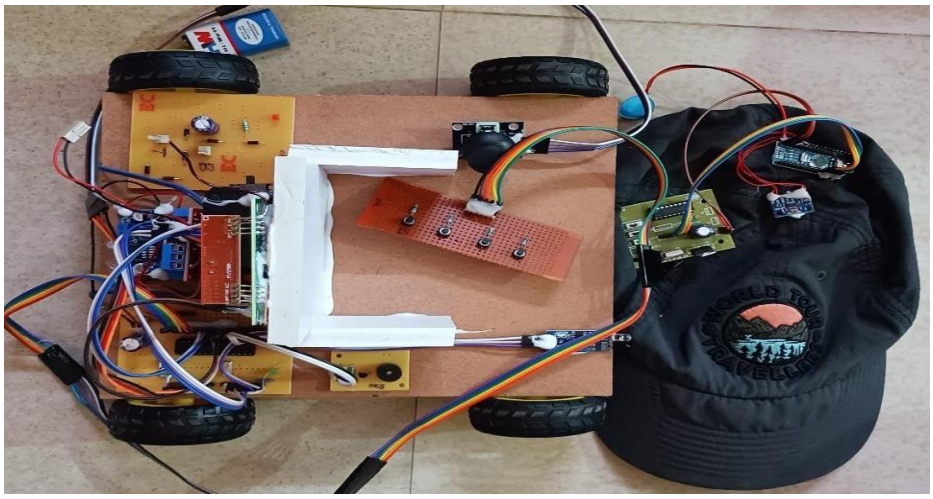

Fig 2: Advanced and Intelligent Wheel Chair equipped with MEMS Sensor and

The **"Advanced and Intelligent Wheelchair Control System "** project utilizes a 16F72 Microcontroller to provide enhanced mobility and safety for leg amputees. The system allows users to control the wheelchair with an intuitive joystick, offering precise navigation in various environments. It features an obstacle detection system that alerts users via a Buzzer and automatically stops the wheelchair when obstacles are detected, ensuring safety. The design is energy- efficient, cost-effective, and user-friendly, allowing for longer battery life and easy operation. Test users reported improved independence and confidence, and the system's scalability makes it adaptable to different wheelchair models, with potential for further integration into smart environments and autonomous navigation.

6-CONCLUSION

The "Advanced and Intelligent Wheelchair Control System " project successfully demonstrates a significant advancement in assistive technology for individuals with mobility impairments. By utilizing a 16F72 Microcontroller, the system provides precise and intuitive joystick control, allowing users to navigate wheelchairs easily in various environments. The integration of obstacle detection with automatic stopping and alert mechanisms enhances safety, ensuring that users can avoid potential accidents while moving.

The system's energy-efficient design and cost-effective components make it accessible and practical for a wide range of users. Furthermore, the project's positive results indicate its potential to improve the independence and quality of life for leg amputees and individuals with limited mobility. The successful operation of the wheelchair in real-world tests demonstrates its usability and reliability in everyday settings, from homes to public spaces.

This project lays the groundwork for future enhancements, such as autonomous navigation, smart home integration, and further safety features, which can make mobility even more accessible and efficient. Ultimately, the system offers a promising solution for empowering people with disabilities, providing them with greater freedom, safety, and confidence in their daily lives.

REFERENCES

[1] Jain, A., & Gupta, M. (2018). "Design and Development of Assistive Technology for Disabled People Using IoT." International Journal of Engineering and Technology (IJET). This paper discusses the development of assistive technologies, including wheelchairs, and explores the integration of IoT for improving mobility for disabled individuals.

[2] Rani, S., & Kumar, R. (2020). "Wheelchair Automation System with Obstacle Detection and Voice Control." Journal of Robotics and Automation (JRA). This article covers the use of various control mechanisms, including voice control, for enhancing wheelchair mobility and safety, with a focus on automation and obstacle detection.

[3] Zhou, L., & Wang, Z. (2017). "Design of Intelligent Wheelchair System Using Microcontroller." International Journal of Robotics and Automation (IJRA). This paper provides insights into the design and implementation of intelligent wheelchair systems, discussing the microcontroller-based control and sensor integration for autonomous navigation and safety features.

[4] Pustokhina, I., & Novikova, A. (2016). "Design of an Autonomous Wheelchair for Disabled People with Obstacle Detection." Proceedings of the International Conference on Robotics and Automation. This reference discusses advanced wheelchair systems with obstacle detection, including the use of sensors and intelligent algorithms for navigating complex environments.

[5] Patel, A., & Dey, S. (2019). "Joystick-Controlled Wheelchair with Obstacle Avoidance System." Journal of Engineering and Technology (JET). This article explores the design and implementation of joystick-controlled wheelchairs with obstacle avoidance and safety mechanisms, which aligns with the functionality of the project described.

[6] Amirabdollahian F., Ates S., Basteris A., et al. Design, development and deployment of a hand/wrist exoskeleton for home-based rehabilitation after stroke - SCRIPT project. Robotica. 2019.

[7] Mónica Faria B., Vasconcelos S., Paulo Reis L., Lau N. Evaluation of distinct input methods of an intelligent wheelchair in simulated and real environments: a performance and usability study. Assistive Technology. 2018;25(2):88–98.

[8] Urdiales C., Peula J., Barrue C., et al. A new multi-criteria optimization strategy for shared control in wheelchair assisted navigation. Autonomous Robots. 2018;30(2):179–197.