IOSR Journal of Engineering (IOSRJEN) **www.iosrjen.org**
ISSN (e): 2250-3021, ISSN (p): 2278-8719
PP 25-31

Utilization Of Zigbee Module In Rssi Localizationapplication Inside Prison

Gokula Brindha.B[1] , Arul Jothi.S[1], Prof. Saranya.S[2]
[1](*BE Student , Department of ECE, DSCET, Anna university, India)*
[2](*Assistant profesor, Department of ECE, DSCET, Anna university,,India)*

Abstract: *Existing researches on location tracking focus either entirely on indoor or entirely on outdoor by using different devices and techniques. Several solutions have been proposed to adopt a single location sensing technology that fits in both situations. This paper aims to track a user position in both indoor and outdoor environments by using a single wireless device with minimal tracking error. RSSI (Received Signal Strength Indication) technique together with enhancement algorithms is proposed to cater this solution. The proposed RSSI-based tracking technique is divided into two main phases, namely the calibration of RSSI coefficients (deterministic phase) and the distance along with position estimation of user location by iterative trilateration (probabilistic phase). A low complexity RSSI smoothing algorithm is implemented to minimize the dynamic fluctuation of radio signal received from each reference node when the target node is moving. Experiment measurements are carried out to analyze the sensitivity of RSSI. The results reveal the feasibility of these algorithms in designing a more accurate real-time position monitoring system.*
Keywords: *Global Robot localisation,Kalman filtering radio frequency identification (RFID) Technology.*

I. Introduction

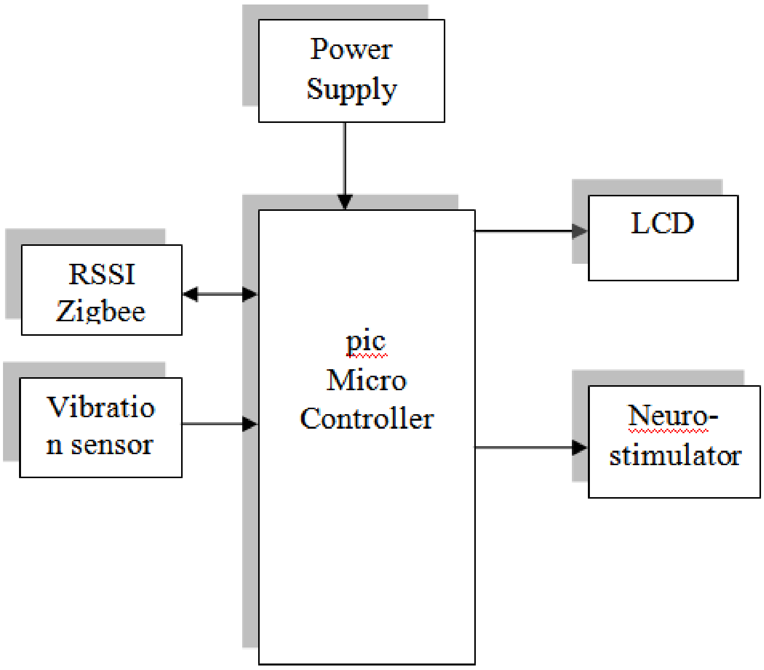
RSSI localizationhave recently emerged as a premier research topic. They have great long term economic potential, ability to transform our lives, and pose many new system-building challenges. Sensor networks also pose a number of new conceptual and optimization problems, some of these such as location, deployment, and tracking, are fundamental issues, in that many applications rely on them for needed information. Coverage in general, answers the questions about quality of service (surveillance) that can be provided by a particular sensor network. The integration of multiple types of sensors such as seismic, acoustic, optical, etc. in one network platform and the study of the overall coverage of the system also presents several interesting challenges.

II. Comparison Of Existing And Proposed System

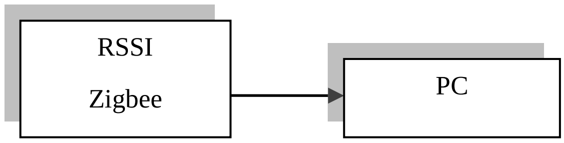
Existing System	**Proposed System**
➢ In this system, there is CCTV device to monitor the localization of the prisoner inside the jail. ➢ Need to review the camera continuously until the patient is tracked. **Drawbacks** ➢ Targets may escape in the CCTV camera. ➢ High cost.	➢ In this system, RSSI zigbee is used to track the prisoner location at any time. ➢ The RSSI zigbee technology helps in reading the prisoner which tells the current location. **Advantages** ➢ Fast response. ➢ Low cost and wireless.

2.1 Proposed System:

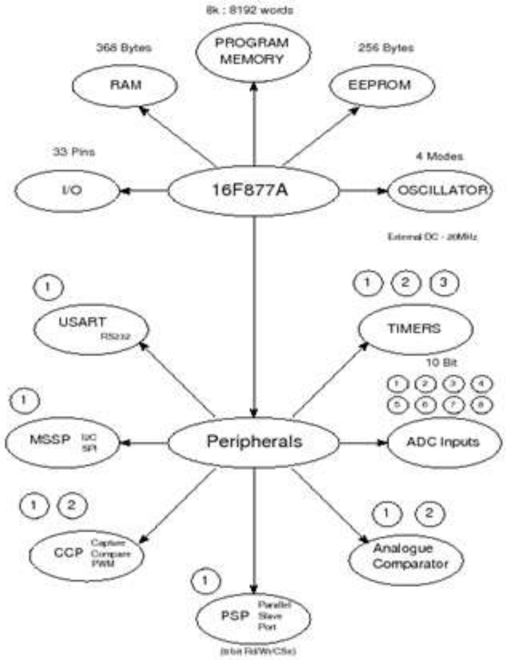
In this paper, an approximate initial information on the position of the tags is required, even if errors up to about 1 mcan be tolerated by the algorithm. A more difficult problem, however, arises if at the beginning, the tag coordinates are completely unknown and must be estimated together with the robot pose. This kind of situation is known in the literature by simultaneous localization and mapping (SLAM). Compared with the localization literature, only a few papers deal with SLAM using the RFID technology. As pointed out in , the limited accuracy in RFID data excludes the possibility of applying well-established SLAM approaches in this context. This is particularly true when RFID tags are used as binary sensors. In this case, the SLAM is faced with specific approaches or assumptions [15]–[18]. Using the time response of RFID tags, applied an EKF (properly initialized) to solve a SLAM problem and achievesa localization accuracy in the order of about 0.5 m.



2.2 Overall diagram
Receiver section:section 1 (Prisoner1):



Transmitter Section 2 (control room):



2.3 Proposed system explanation:

In the proposed system, we use RSSI technology to monitor the location and position of prisoner will always hold the RSSI Zigbeeand neuron stimulator. For example, if the prisoner is try to move outside the jail it automatically indicate through buzzer in the control room based on the signal strength. The control room automatically sends a control to activate neuron simulator. If the prisoner is try to break the module is sense by vibration sensor, it will automatically produce alarm in the prisoner sect and control section. The RSSI ZIGBEE inside the room reads the location and distance to the control section. In the receiver section, the data is gathered and stored in the PC for the future use. LCD will display the location and action performed by the microcontroller.

2.4 Hardware Requirments And Explanation Modules

1. **Controller Module**
2. **Rssi Module**
3. **Sensor Module**
4. **Auxiliary Modules**

- PIC Microcontroller
- AT89c51 microcontroller
- RSSI zigbee
- Neuro-stimulator
- LCD
- Buzzer
- Vibration sensor

2.5 Controller Module (Pic Microcontroller)

III. Introduction Of Pic16f877a:

The PIC16F877A CMOS FLASH-based 8-bit microcontroller is upward compatible with the PIC16C5x, PIC12Cxxx and PIC16C7x devices. It features 200 ns instruction execution, 256 bytes of EEPROM data memory, self programming, an ICD, 2 Comparators, 8 channels of 10-bit Analog-to-Digital (A/D) converter, 2 capture/compare/PWM functions, a synchronous serial port that can be configured as either 3-wire SPI or 2-wire I2C bus, a USART, and a Parallel Slave Port.

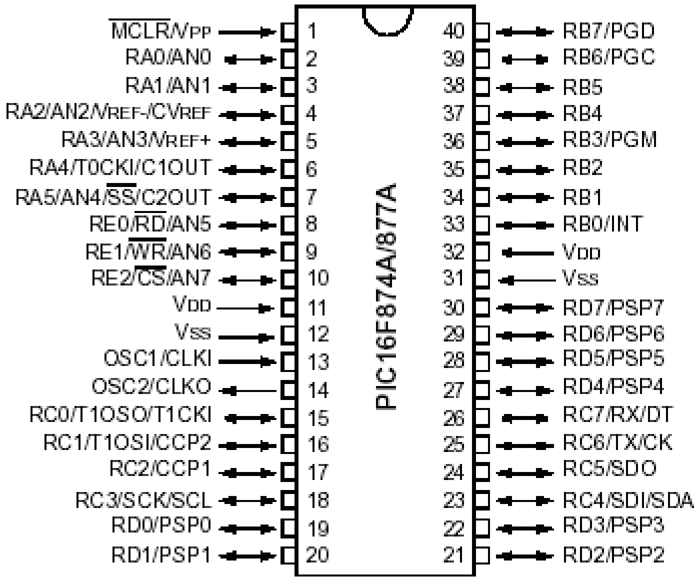
3.1 Microchip PIC16F877A Microcontroller Features

3.1.1 High-Performance RISC CPU

- Operating speed: 20 MHz, 200 ns instruction cycle
- Operating voltage: 4.0-5.5V
- Industrial temperature range (-40° to +85°C)
- 15 Interrupt Sources
- 35 single-word instructions
- All single-cycle instructions except for program branches (two-cycle)

Special Microcontroller Features

- Flash Memory: 14.3 Kbytes (8192 words)
- Data SRAM: 368 bytes
- Data EEPROM: 256 bytes
- Self-reprogrammable under software control
- In-Circuit Serial Programming via two pins (5V)
- Watchdog Timer with on-chip RC oscillator
- Programmable code protection
- Power-saving Sleep mode
- Selectable oscillator options
- In-Circuit Debug via two pins



3.2 Peripheral Features

- 33 I/O pins; 5 I/O ports
- Timer0: 8-bit timer/counter with 8-bit prescaler
- Timer1: 16-bit timer/counter with prescaler
    - Can be incremented during Sleep via external crystal/clock
- Timer2: 8-bit timer/counter with 8-bit period register, prescaler and postscaler
- Two Capture, Compare, PWM modules
    - 16-bit Capture input; max resolution 12.5 ns
    - 16-bit Compare; max resolution 200 ns
    - 10-bit PWM
- Synchronous Serial Port with two modes:
    - SPI Master
    - I2C Master and Slave
- USART/SCI with 9-bit address detection
- Parallel Slave Port (PSP)
    - 8 bits wide with external RD, WR and CS controls
- Brown-out detection circuitry for Brown-Out Reset

3.3 Analog Features

- 10-bit, 8-channel A/D Converter
- Brown-Out Reset
- Analog Comparator module
    - 2 analog comparators
    - Programmable on-chip voltage reference module
    - Programmable input multiplexing from device inputs and internal VREF
    - Comparator outputs are externally accessible

3.4 Memory of the PIC16F877 divided into 3 types of memories:

- **Program Memory**- A memory that contains the program(which we had written), after we've burned it. As a reminder, Program Counter executes commands stored in the program memory, one after the other.
- **Data Memory**– This is RAM memory type, which contains a special registers like SFR (Special Faction Register) and GPR (General Purpose Register). The variables that we store in the Data Memory during the program are deleted after we turn of the micro.
  These two memories have separated data buses, which makes the access to each one of them very easy.
- **Data EEPROM (Electrically Erasable Programmable Read-Only Memory)** - A memory that allows storing the variables as a result of burning the written program.

IV. PIC16F87XA Program Memory

The PIC16F87XA devices have a 13-bit program counter capable of addressing an 8K word x 14 bit program memory space. This memory is used to store the program after we burn it to the microcontroller. The PIC16F876A/877A devices have 8K words x 14 bits of Flash program memory that can be electrically erased and reprogrammed. Each time we burn program into the micro, we erase an old program and write a new one.

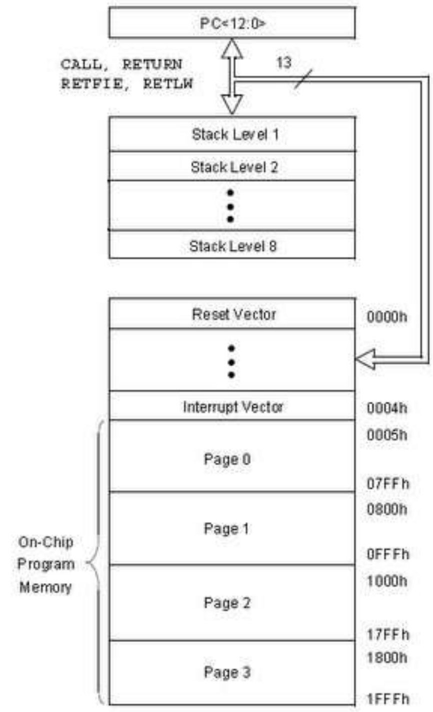


fig:-PIC16F876A/877A program memory map and stack

Program Counter (PC) keeps track of the program execution by holding the address of the current instruction. It is automatically incremented to the next instruction during the current instruction execution.

The PIC16F87XA family has an 8-level deep x 13-bit wide hardware stack. The stack space is not part of either program or data space and the stack pointer is not readable or writable. In the PIC microcontrollers, this is a special block of RAM memory used only for this purpose.

The CALL instruction is used to jump to a subroutine, which must be terminated with the RETURN instruction. CALL has the address of the first instruction in the subroutine as its operand. When the CALL instruction is executed, the destination address is copied to the PC. The PC is PUSHed onto the stack when a CALL instruction is executed, or an interrupt causes a branch. The stack is POP'ed in the event of a RETURN, RETLW or a RETFIE instruction execution.

The stack operates as a circular buffer. This means that after the stack has been PUSHed eight times, the ninth push overwrites the value that was stored from the first push. The tenth push overwrites the second push (and so on).

Each time the main program execution starts at address 0000 - Reset Vector. The address 0004 is "reserved" for the "interrupt service routine" (ISR).

If we plan to use an interrupt, our program will begin after the Interrupt Vector; and if not we can start to write from the beginning of the Reset Vector.

4.1 Rssi Module

- **Rssi Zigbee**

The **ZigBee (cc2530)** is a true system on chip (SoC) solution for IEEE 802.15.4 applications. It combines the excellent performance of a leading RF transreceiver with an industry-standard enhanced 8051 MCU, in system programmable flash memory, 8 kB RAM, and many other powerful features. Received Signal Strength Indicator (RSSI) is a measurement of power present in a received radio signal.

Features
• Supply voltage: 5v DC
• RS232 output
• Output power: up to 4.5dBm–
• Detection range: (10-20) m
• Frequency: 2.4GHz
• Ultra-low power consumption

Applications
• Remote control systems
• Home/building automation
• Lighting systems
• Consumer electronics
• Health care

4.2 Sensor Module

➢ **Vibration Sensor**

Sensor is mounted at the bottom of the unit. The unit should be fixed with the vibrating body firmly the sensitivity is adjusted for the required vibration/ shock is detected the output goes low and the delay is provided for proper operation vibrating frequency and amplitude can be detected.

Specification:

- Supply- DC +2v ripple free
- Output current-PNP 100ma
- Analog o/p- 10ma

Sensors available to detect the flame / fire smoke flow level speed position Temp Bio medical application.
Special sensor can be developed against specific requirement.

4.3 Auxiliary Module

Liquidcrystal Displays:
The LCD standard requires 3 control lines and 8 I/O lines for the data bus.

• **RS: Register Select**
RS=0
RS = 1 -> Data Register is selected
• **R/W: Read or Write**
0 -> Write, 1 -> Read
• **E: Enable (Latch data)**
The 8 data lines are connected to PORT 1 of 8051 microcontroller. The three control lines(RS,RW and EN) are connected to PORT 3.5,3.6 and 3.7 respectively.

V. Software Explanation

➢ **Embedded C:**

As time progressed, use of microprocessor-specific assembly-only as the programming language reduced and embedded systems moved onto C as the **embedded programming language** of choice. C is the most widely used programming language for embedded processors/controllers. Assembly is also used but mainly to implement those portions of the code where very high timing accuracy, code size efficiency, etc. are prime requirements.

Initially C was developed by Kernighan and Ritchie to fit into the space of 8K and to write (portable) operating systems. Originally it was implemented on UNIX operating systems. As it was intended for operating systems development, it can manipulate memory addresses. Also, it allowed programmers to write very compact codes. This has given it the reputation as the language of choice for hackers too.

As assembly language programs are specific to a processor, assembly language didn't offer portability across systems. To overcome this disadvantage, several high level languages, including C, came up. Some other languages like PLM, Modula-2, Pascal, etc. also came but couldn't find wide acceptance. Amongst those, C got wide acceptance for not only embedded systems, but also for desktop applications. Even though C might have lost its sheen as mainstream language for general purpose applications, it still is having a strong-hold in embedded programming. Due to the wide acceptance of **C in the embedded systems**, various kinds of support tools likecompilers & cross-compilers, ICE, etc. came up and all this facilitated development of **embedded systems using C**.

5.1 Components Of Mplab Ide:

The MPLAB IDE has both built-in components and plug-in modules to configure the system for a variety of software and hardware tools.

5.2Debugger:

The Microchip debugger allows breakpoints, single stepping, watch windows and all the features of a modern debugger for the MPLAB IDE. It works in conjunction with the editor to reference information from the target being debugged back to the source code.

VI. Conclusions

The RSSI and the phase can be considered two redundant measurements of the same quantity: the distance of the robot from the projection of the tag on the floor. However, the information delivered by these two quantities presents different features, for some aspects complementary: phase measurements are more sensitive than the RSSI to a change in the robot position but suffer from a cycle ambiguity, which complicates matters. The optimal combination of the two measurements is not straightforward and represents a challenging problem,dual in a sense to the problem of the dynamic allocation of redundant actuators. An effective solution is proposed in this paper by considering an algorithm characterized by two stages: in the first stage, only the RSSI is used for estimation purposes, while in the second stage, also the phase measurements are considered. A dynamic criterion, based on the observation of the variance of the estimation of the tag coordinates, is adopted to decide the switching time between the two stages, which is performed independently on each tag. The proposed two-stage algorithm estimates the robot pose and simultaneously improves the estimate on the tag coordinates.Errors up to about 1 m can be tolerated by the algorithm on the initial estimate of the tag positions. If no information is available at the beginning on the tag coordinates, range only SLAM techniques could be considered to solve the problem, capitalizing on the ideas presented in this paper to properly combine RSSI and phase measurements.

Reference

[1] R. Tesoriero, J. A. Gallud, M. D. Lozano, and V. M. R. Penichet, “Tracking autonomous entities using RFID technology,” *IEEE Trans. Consum.Electron.*, vol. 55, no. 2, pp. 650–655, May 2009.

[2] M. Boccadoro, F. Martinelli, and S. Pagnottelli, “Constrained and QuantizedKalman filtering for an RFID robot localization problem,” *Auto. Robots,* vol. 29, nos. 3–4, pp. 235–251, 2010.

[3] B.-S. Choi, J.-W.Lee, J.-J.Lee, and K.-T. Park, “A hierarchical algorithm for indoor mobile robot localization using RFID sensor fusion,”

[4] *IEEE Trans. Ind. Electron.*, vol. 58, no. 6, pp. 2226–2235, Jun. 2011.

[5] P. Yang, W. Wu, M. Moniri, and C. C. Chibelushi, “Efficient object localization using sparsely distributed passive RFID tags,” *IEEE Trans.*

[6] *Ind. Electron.*, vol. 60, no. 12, pp. 5914–5924, Dec. 2013.

[7] E. Di Giampaolo, and F. Martinelli, “A passive UHF-RFID system for the localization of an indoor autonomous vehicle,” *IEEE Trans. Ind. Electron.*, vol. 59, no. 10, pp. 3961–3970, Oct. 2012.
Bekkali, H. Sanson, and M. Matsumoto, “RFID indoor positioning based on probabilistic RFID map and Kalman filtering,” in *Proc. 3rd IEEE Int. Conf. Wireless Mobile Comput .,Netw. Commun.*, Oct. 2007, p. 21.

[8] S. S. Saab and Z. S. Nakad, “A standalone RFID indoor positioning system using passive tags,” *IEEE Trans. Ind. Electron.*, vol. 58, no. 5, pp. 1961–1970, May 2011.

[9] S. Park and H. Lee, “Self-recognition of vehicle position using UHF passive RFID tags,” *IEEE Trans. Ind. Electron.*, vol. 60, no. 1, pp. 226–234, Jan. 2013.

[10] E. Di Giampaolo and F. Martinelli, “Mobile robot localization using the phase of passive UHF RFID signals,” *IEEE Trans. Ind. Electron.*, vol. 61, no. 1, pp. 365–376, Jan. 2014.

[11] S. Sarkka, V. V. Viikari, M. Huusko, and K. Jaakkola, “Phase-based UHF RFID tracking with nonlinear Kalman filtering and smoothing,” *IEEE Sensors J.*, vol. 12, no. 5, pp. 904–910, May 2012.