3WD Omni-Wheeled Mobile Robot using ARM Processor for Line Following Application

Mohammad Afif Bin Ayob
Dept. of Mechatronics and Robotics Eng.
Fac. of Electrical and Electronics Eng.
Universiti Tun Hussein Onn Malaysia (UTHM)
86400 Batu Pahat, Johor, Malaysia
AF11F@yahoo.com

Mohamad Fauzi Zakaria
Dept. of Mechatronics and Robotics Eng.
Fac. of Electrical and Electronics Eng.
Universiti Tun Hussein Onn Malaysia (UTHM)
86400 Batu Pahat, Johor, Malaysia
mfauzi@uthm.edu.my

Abstract—**Performance efficient processors have become increasingly important for mobile robots with numerous actuators and sensors. Therefore, this project is to implement a 32-bit low-power ARM processor into the development of a 3WD omni-directional mobile robot equipped with IR sensor capabilities. The scope of the project is narrowed down to the usage of Embedded Artists' LPC2148 USB QuickStart Board, digital fiber sensor, DC brushless motor, and omni-wheel. Most of the tools and necessary equipment for the project development can be obtained within the UTHM. At the end of this project, the ARM processor has successfully generate the desired PWM for motor control and respond to the digital input from the interface of IR sensor to the microcontroller.**

Keywords - 3D omni-wheel; mobile robot; ARM processor

I. Introduction

ARM processors chip that contains two 16 Kbyte, 32-way set associative caches for instructions and data can be used in mostly any domain such as handheld devices, automation system, robotics, and other consumer electronics since it provides comprehensive support required in developing a complete system [1]. Compared to the traditional PIC microcontroller that is based on 8-bit architecture, has small RAM, slow processing speed and relatively provides low I/O pins, ARM processor is chosen mostly because it delivers all the opposite characteristics of the PIC microcontroller, offers high speed data processing, and optimizes power consumption since it is a growing problem in deep submicron digital circuit design [2].

The frequent uses of conventional type of wheel in mobile robot development only because of its simplicity and availability can only offer a single rotational axis in which the movement is restricted to the axle shaft it is attached to. As omnidirectional vehicles have high flexibility and high motion performance the replacement of the conventional system with omniwheels made it the best solution for the problem since it can provide multi-directional and sideways movement perpendicular to the wheel's rotation [3]. The wheel driving system of the mobile robot plays a vital role in allowing the robot to navigate along a limited direction depending on its wheel architecture [4].

Mobile robots that use timers and delays instead of sensors for line navigation mode doesn't always produce the accurate and precise movement. The implementation of IR sensors in this scope of the project will further enhance and improve the existing system. In addition, infrared sensors are also largely used in obstacle avoiding processes because they are fast and cheap, and require only simple signal processing [5]. The implementation of this sensor is practically ideal because sensor systems for mobile robots must usually be relatively small, lightweight, and inexpensive [6].

The objectives of this project are as follows:

1. To design and develop a 3-wheel drive mobile robot by using 32-bit low-power ARM processor instead of traditional 8-bit PIC microcontroller.
2. To explore functionality and the system architecture of ARM processor.
3. To implement a line follower application to a mobile robot by using IR sensor.
4. To design a mechanical system of a robot that uses three independent omniwheels for its movement.

The mobile robot in this project possessed a perfect form of triangle positioning for the motor drive at an angle of 60 from each other and consequently a proper ratio for the speed of the motor is needed for the motor control.

This paper will further put the discussion of the project in details regarding the methodology, result and analysis, and the conclusion of this project.

II. Methodology

A. Project Activities

The project starts by first identifying the problems that exist in the current system of mobile robot. A literature review was done on related knowledge to assist in any ways that it may. Such reviews are based on international publications, websites, and engineering books. Detail research in hardware is needed for the robot mechanical development while at the same time a lot of time has been spent to search and compare for compatible ARM compiler in terms of availability, performance and technical supports. The system requirement was then determined to do this project.

The next step is followed by designing the hardware and software part of the robot system wherein both the hardware design and the software design will be interfaced together to see whether the system works. Any malfunction that occurred was resolve. Overall system testing need to be done afterwards in order to verify the functionality of both basic and application program in this project.

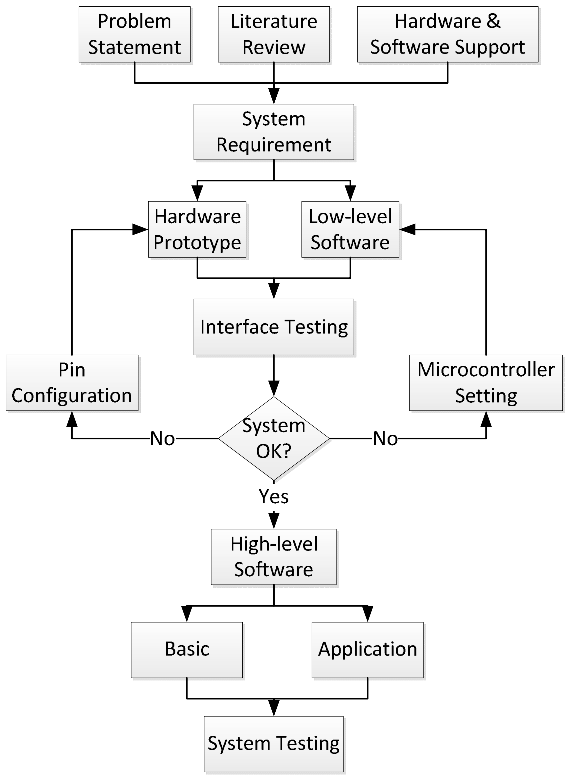


Figure 1: Flowchart of the project activities.

B. System Architecture

The LPC2148 USB QuickStart Board have a total of 25 connected I/O pins; 9 pins for Alphanumeric LCD (for data bus including control and command line), 3 pins for IR sensors, 9 pins for three VEXTA motors (three pins each), and 4 pins for active-low SPST push buttons.

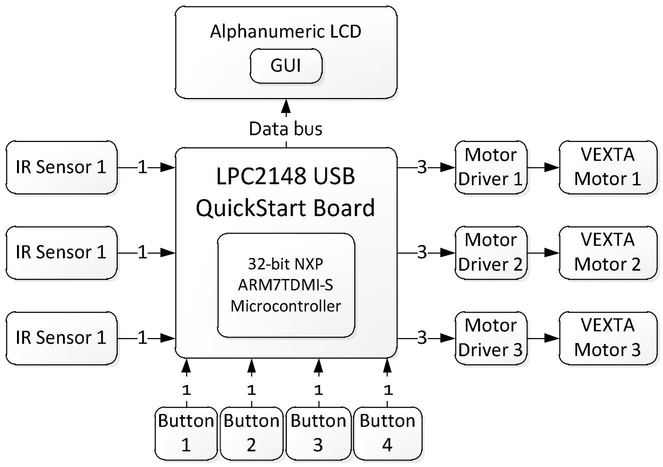


Figure 2: Block diagram of LPC2148 connected I/O pins.

C. Types of Compiler

The ARM C compilers between GNU C Compiler, KEIL uVision3, and IAR Embedded Workbench were benchmarked by Sylvia Gomes Augusto and Lionel Orry [7]. This benchmark compares three C compilers for ARM:

- GNU – GNU C Compiler for ARM, version 4.0
- IAR – 32K code-size limited version of the C Compiler delivered with the Embedded Workbench, version 4.20A
- KEIL – C Compiler delivered with µVision3, version 3.12a

FEATURES	GNU	KEIL	IAR	BEST
Code Size (no FP)	1 (4464.5)	1.5 (6365)	1 (4254.8)	IAR
Speed (no FP)	1 (527.9)	1.8 (975.7)	1.1 (596.1)	**GNU**
Code Size (with FP)	1 (2200)	NA	1 (2216)	**GNU**
Speed (with FP)	1 (0.9)	NA	2 (1.8)	**GNU**

Table 1: Benchmark comparisons for ARM C Compiler.

The benchmark shows that GNU C Compiler yields the best results and hence is the best choice in terms of code size and speed optimizations. It should be noted that the GNU C Compiler can be obtained for free and is supported by the open source community while KEIL and IAR worth $2895 for basic edition and 2150EUR for baseline edition respectively.

Hence, from these findings, the latest WinARM of version 20060606 that includes Programmers Notepad Editor version 2.0.6.1 and GNU C/C++ Compiler version 4.1.1 is used to program the LPC2148 USB QuickStart Board. The tools also said should work with any microcontroller with ARM architecture.

D. Hardware

The complete design and size of the robot was made by using SolidWorks 2009. The schematic and PCB design of the electrical circuits were made by using ISIS and ARES from Proteus Professional v7.5. The circuits were then fabricated in double layer at the Printed Circuit Board (PCB) Laboratory. The overall circuits design is given in Appendix 1 for Main Board, Appendix 2 for Motor Board, and Appendix 3 for LCD Board.

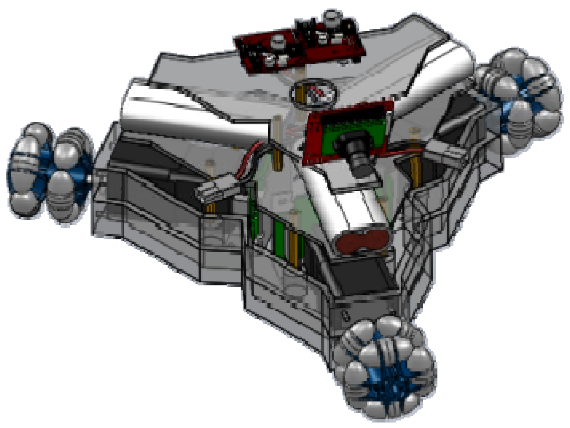


Figure 3: Isometric view of the robot designed in SolidWorks 2009.

To configure the mobile robot as Figure 4, to move either forward/backward, only Motor 1 and Motor 2 need to be controlled in counterclockwise (CCW)/clockwise (CW) for both motors while the speed for Motor 3 should be set to 0. To move the mobile robot to the left, the speed ratio and the direction that the motors should spin should be divided as:

$$\text{Motor 1} : \text{Motor 2} : \text{Motor 3} = 1 : 2 : 3 = \text{CW} : \text{CCW} : \text{CW}$$

The same rules applied to move the mobile robot to the right.

$$\text{Motor 1} : \text{Motor 2} : \text{Motor 3} = 2 : 1 : 3 = \text{CCW} : \text{CCW} : \text{CW}$$

To make the mobile robot to turn left or right, all three motors should be set at the same speed and the rotational direction is also the same; CW for turning left and CCW for turning right.

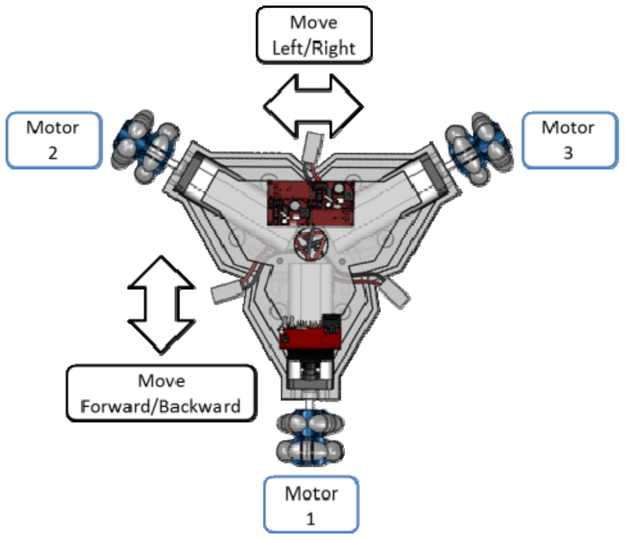


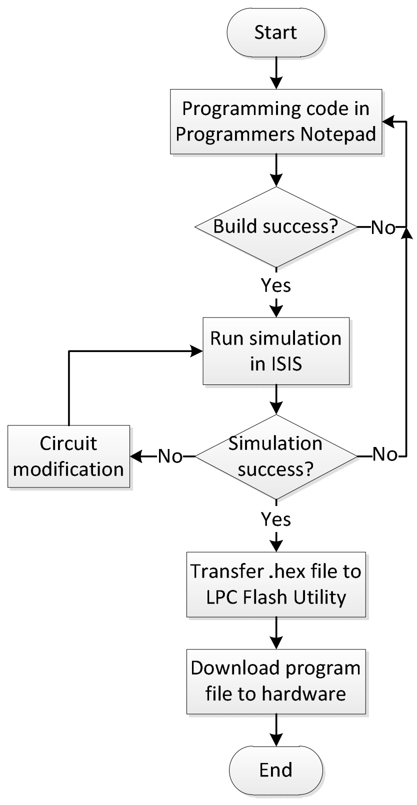
Figure 4: Configuration of robot movement.

E. Software

The program file to be downloaded to the microcontroller was written in Programmers Notepad and was simulated in ISIS. The hex file that was produced from compiling the program was then transferred serially to the LPC2148 USB QuickStart Board via a USB to RS232 converter cable using LPC Flash Utility version 2.2.3 that can be obtained for free from NXP website. Figure 5 shows the overall flow process.

The IR sensor will first initialize all variables included and will detect line afterwards as shown in Figure 6. The sensor will either initiate or stop the motor movement in Motor Control subroutine and continuously repeating the same steps corresponding on the line it detects.

The Pulse Width Modulation (PWM) of the Motor Control subroutine in Figure 7 is first initialized before the readings from the IR Sensor subroutine act as the trigger input to drive the corresponding wheels. The program will continuously wait for another reading from the IR sensors.


Start
Programming code in Programmers Notepad
Build success?
No
Yes
Run simulation in ISIS
Circuit modification
No
Simulation success?
No
Yes
Transfer .hex file to LPC Flash Utility
Download program file to hardware
End
Figure 5: Process taken to program .hex file to microcontroller.

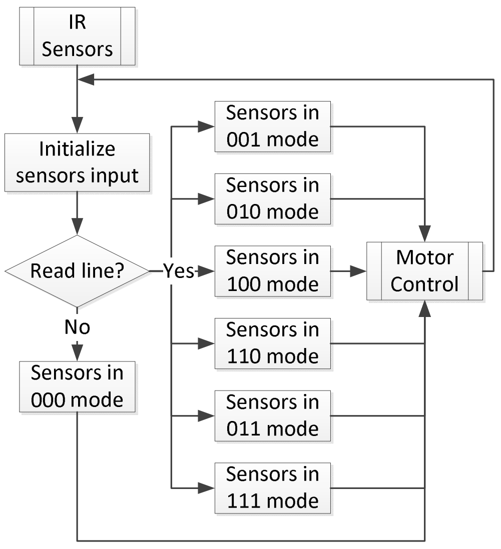


Figure 6: Low-level software for IR sensor.

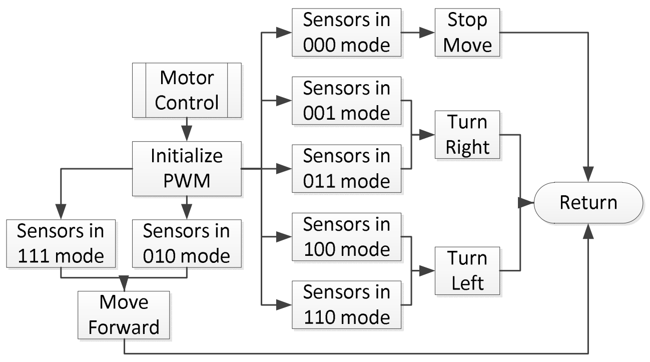


Figure 7: Low-level software for motor control.

The application of the high-level software in Figure 8 below starts by configuring the clock frequency of the microcontroller to 12MHz, Phase-Locked Loop (PLL) Multiplier to 5, Peripheral Bus Speed Divider (PBSD) to 4, and not to forget disabling the watchdog timer. The corresponding I/O ports were then initialized before the IR Sensor subroutine takes place.

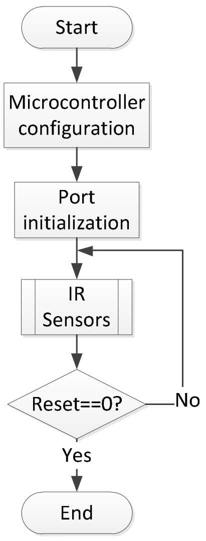


Figure 8: High-level software application.

III. RESULT AND ANALYSIS

A. Hardware

The main body of the robot as in Figure 9 was constructed by using various sizes of thick transparent acrylic glass at the Computer Numerical Control (CNC) Laboratory. The robot was made within 42cm squared area and 22cm height for practicality and portability.

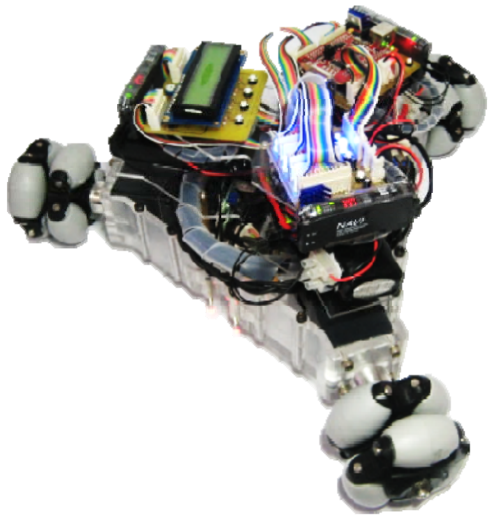


Figure 9: Mechanical outcome of the mobile robot.

Figure 10 below shows the Main Board that contains the LPC2148 USB QuickStart Board with connected I/O pins and +12V DC supply. The red indicator LED will lights up whenever the supply is served to the board. The LPC2148 is powered by +5V DC supply that is being regulated from the LM7805 voltage regulator. The Main Board also has a push button to reset the program whenever it is needed.

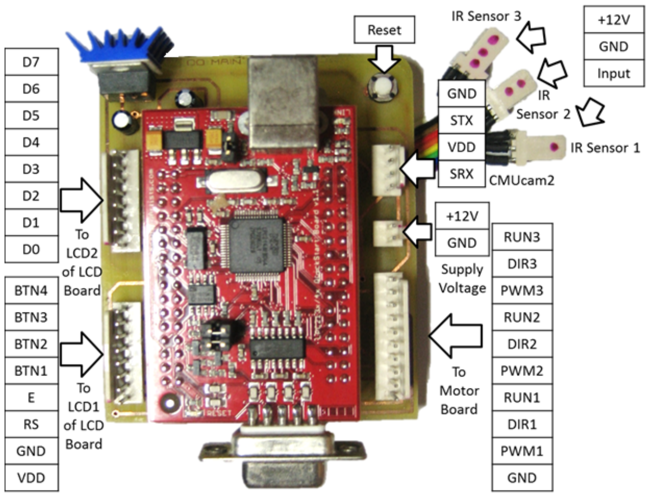


Figure 10: Main Board circuit with pin assignment.

The Motor Board in the following Figure 11 provides motor control up to three VEXTA DC brushless motors. The supply power needs two +12V DC supply that connects serially to serve the +24V DC supply to run the motor drivers. One LED to indicate whether the board is being supplied or not is located at the top right of the board while the all the three VEXTA motors have their own LED indicator next to the supply input.

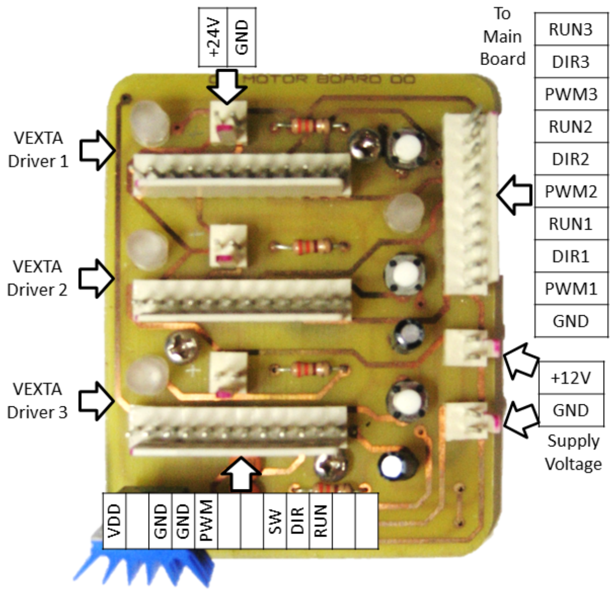


Figure 11: Motor Board circuit with pin assignment.

B. Software

For PWM motor control, it is widely known that the frequency, f is equal to the inverse of period, T.

$$Frequency, f = \frac{1}{T} \tag{1}$$

So for the period of 0.1s (equivalent to 100ms), the resulted frequency is given as:

$$f = \frac{1}{100ms}$$

$$f = 10Hz$$

By setting the PWM duty cycle to 70% and frequency to 10Hz, the simulation output of the graph is given in Figure 12. The output graph proves the exact amount of period and duty cycle, which is 100ms and 70ms respectively.

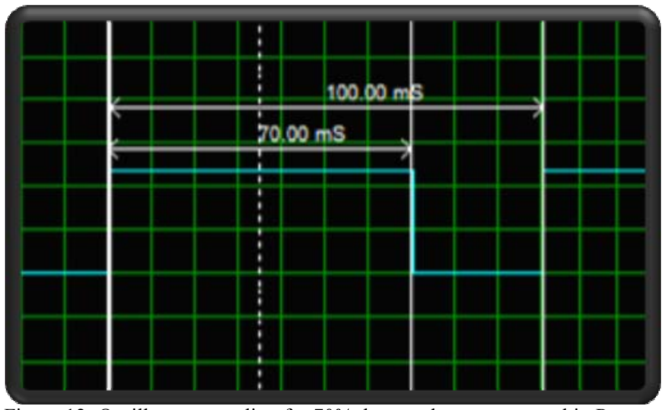


Figure 12: Oscilloscope reading for 70% duty cycle motor control in Proteus Professional v7.5 (simulation).

By downloading the same tested program that contains the 70% PWM duty cycle to the microcontroller, the graph reading is given in Figure 13.

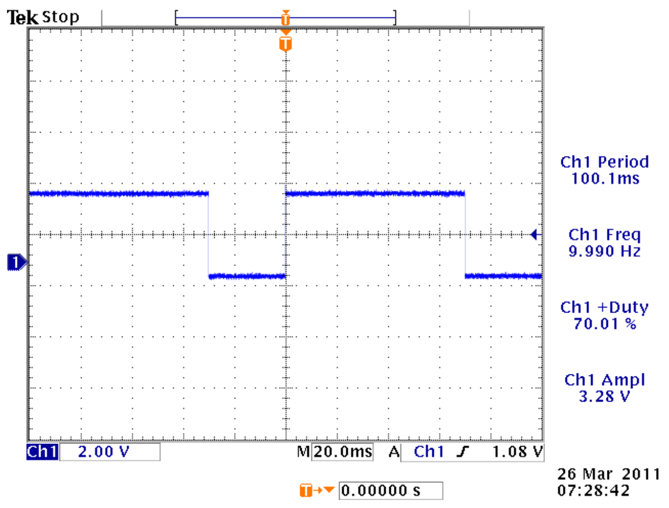


Figure 13: Oscilloscope reading for 70% PWM duty cycle motor control on Tektronix TDS 3032B digital phosphor oscilloscope (hardware implementation).

The comparison on both Figure 12 and Figure 13 above shows the accuracy and functionality of the tested program on real-life application. The IR sensors on the other hand will produce logic 1 that equals to +3.30V if it detects dark line on Light-ON mode and logic 0 that equals to +0.01V if it detects bright line on the same mode.

C. Specification

Microcontroller		NXP ARM7TDMI-S LPC2148
Power Supply		LiPo Rechargeable Battery 11.1V 2200mAH
Sensor		SUN FX-301 Digital Fiber Sensor
Motor		BLH015K-XX* VEXTA DC Brushless Motor
Drive Wheel		Omniwheel
Robot	Weight	2.5kg
	Size	42cm X 42cm X 22cm

Table 2: Specification of the mobile robot.

IV. CONCLUSION

A. Conclusion

The system testing that concerning on the basic and real-time execution has successfully verified the interface functionality between the motor control, IR sensors, and NXP's ARM7TDMI LPC2148 microcontroller. Based on the simulations and tested results, the objectives have been successfully accomplished since the motor can be controlled in specific PWM duty cycle and interfacing with the IR sensors works perfect.

B. Recommendation

This project can be further expanded and enhanced by swapping the IR sensors to CMUcam2 visual sensor in which it can not only be used for line navigation but also for color tracking, frame differencing, and motion sensing. Graphical LCD can also be connected with the remaining I/O ports on the microcontroller to display menu selection and program option. The drive system in the mobile robot can be changed to transwheel that has the same characteristics but better surface coverage that can lead to better robot movement. The use of range finder sensor for obstacle avoidance can also be implemented since the microcontroller features up to 14 dedicated analog to digital (ADC) inputs.

V. REFERENCES

[1] R. Witek, and J. Montanaro, "StrongARM: a high-performance ARM processor," COMPCON Spring '96 - 41st IEEE International Computer Conference, pp. 188, 1996

[2] R. Bai, S. Kulkarni, W. Kwong, A. Srivastava, D. Sylvester, and D. Blaauw, "An implementation of a 32-bit ARM processor using dual power supplies and dual threshold voltages," Proc. IEEE Computer Society Annual Symposium on VLSI, p. 1, 2003.

[3] Ishida, and S. Miyamoto, H., "Ball wheel drive mechanism for holonomic omnidirectional vehicle," IEEE World Automation Congress (WAC), 2010, p. 1, 2010

[4] Mariappan, M. Choo Chee Wee Vellian, and K. Chow Kai Weng, "A navigation methodology of an holonomic mobile robot using optical tracking device (OTD)," TENCON 2009 - 2009 IEEE Region 10 Conference, p.1, 2010

[5] Navarro, D. Benet, G. Blanes, and F., "Line-based incremental map building using infrared sensor ring," IEEE International Conference on Emerging Technologies and Factory Automation, 2008., p.1, 2008.

[6] J. Borenstein, H.R. Everett, L. Feng, and D. Wehe, "Mobile Robot Positioning & Sensors and Techniques," Invited paper for the Journal of Robotic Systems, Special Issue on Mobile Robots. Vol. 14 No. 4, pp. 231 – 249.

[7] Augusto, S. G., & Orry, L. (2006, February). ARM C Compiler Comparisons. Retrieved on October 5, 2010, from http://www.raisonance.com/arm_benchmark.html.