2008 10th Intl. Conf. on Control, Automation, Robotics and Vision
Hanoi, Vietnam, 17–20 December 2008

A Mobile Robot with a Inter–Integrated Circuit System

Mr. Surachai Panich
Srinakarinwirot University
Ongkarak, Nakornnayok, Thailand
Email: psura44@hotmail.com

Abstract—**This paper will explain a new way to communicate between mobile robots based on PC– Card and some sensors using in robot application or mobile robot research. The hardware interface with the Inter–Integrated Circuit System or shortly I^{2}C bus system will be mainly described here. The mobile robot components, I^{2}C bus system and some I^{2}C slave devices are also introduced. Finally, the software is developed to control sensor data from the robot (master device) to slave devices based on the I^{2}C bus with C^{++}.**

Keywords—**interface system, I^{2}C system, mobile robot**

I. INTRODUCTION

The environment perception of the robot can be accomplished by means of sensors. The sensor data processing can be used connect on the robot by many ways. Nowadays play microcontrollers the important role, because they can be found easily in the electronic shop, and widely developed from many companies, but the Inter-Integrated Circuit (I^{2}C) bus is recently introduced. The I^{2}C bus is an industry standard synchronous serial data communications bus used by many common consumer electronics and embedded systems components. Originally developed by Philips for television tuner integrated circuits, I^{2}C bus interfaces are now found on hundreds of consumer electronics chips, including data converters, EEPROMs, thermal sensors, and real-time clocks [1]. Since I^{2}C is a relatively low bandwidth communications bus, it has found widespread to use in control interfaces in signal processing devices with separate data interfaces. Typically, I^{2}C bus analysis is done with expensive test and measurement equipment, dedicated microcontroller based system, or desktop computer with a dedicated interface card. This paper describes an I^{2}C bus system implemented on the Pioneer II. Section II, explains the major features of Pioneer II robot. In Section III, concepts of the I^{2}C bus are briefly reviewed. Section IV describes the organization of the hardware interface between the Pioneer II robot (as master device) and slave device and overview of the project is given. Section V, Software is developed under Visual C++. Finally, Section VI contains conclusions and ideas for future work.

II. THE ROBOT ARCHITECTURE

In this section, the global hardware architecture is detailed and each module is presented. It consists of two parts, the robot and the addition system (I^{2}C bus system). For the first part, our Laboratory has a Pioneer II [5] - [6] (see figure. 1), a family of mobile robots. They are small, intelligent robots. These are truly off-the-shelf, "plug and play" mobile robots, containing all of the basic components for sensing and navigation in a real-world environment, including battery power, drive motors and wheels, position / speed encoders. They are all managed via an onboard microcontroller and mobile-robot server software. For addition system, the I^{2}C bus is recommended that the can integrate more sensors on the robot. This system will be explained in the next section.

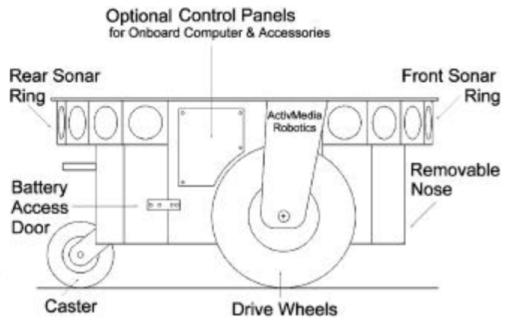

Figure1. Shows overall of the Pioneer II architecture.

A. Main Components of Pioneer II

All Pioneer-II models use a high-performance 20 MHz Siemens 88C166-based microcontroller, with independent motor/power and sonar-controller boards for a versatile operating environment. The controller has two RS232-Standard communications ports and an expansion bus to support the many accessories available for Pioneer, as well as your own custom attachments. And the Pioneer-II comes with high precision (9,850 ticks -per-revolution)wheel-motor encoders for finer odometry, and translation and rotation speed controls. The Pioneer- II also supports a full complement of sixteen sonar(eight fronts and eight rears) for nearly seamless object detection.

B. *Microcontroller*

Pioneer-II's microcontroller uses a 20 MHz Siemens 88C166 microprocessor with integrated 32K flash- ROM. The microcontroller also has 32K of dynamic RAM, two RS232-compatible serial ports, several digital and analog-to-digital, and PSU I/O user-accessible ports, and an eight-bit expansion bus. All the I/O ports, except those used for the motors, encoders, and sonar, are available to the user for Pioneer-II accessory hardware, which you may control through the P2OS.

C. *Motors and Position Encoders*

Pioneer II's drive system uses high-speed, high-torque, reversible DC motors. Each front drive motor includes a high-resolution optical shaft encoder that provides 9,850 ticks per wheel revolution (19 ticks per millimeter) for precise position and speed sensing and advanced dead-reckoning.

III. THE I^{2}C BUS SYSTEM OVERVIEW

When connecting multiple devices to a microcontroller, the address and data lines of each device were conventionally connected individually. This would take up precious pins on the microcontroller, result in a lot of traces on the PCB (Print Circuit Board), and require more components to connect everything together. This made these systems expensive to produce and susceptible to interference and noise. To solve this problem, Philips developed I^{2}C bus, in the 1980s. I^{2}C bus is a low-bandwidth, short distance protocol for on board communications. The name I^{2}C is shorthand for a standard Inter-IC (integrated circuit) bus. I^{2}C bus provides good support for communication with various slow, on-board peripheral devices that are accessed intermittently, while being extremely modest in its hardware resource needs. It is a simple, low-bandwidth, short-distance protocol. Most available I^{2}C bus devices operate at speeds up to 400Kbps, with some venturing up into the low megahertz range. The I^{2}C bus system is easy to use to link multiple devices together since it has a built-in addressing scheme.

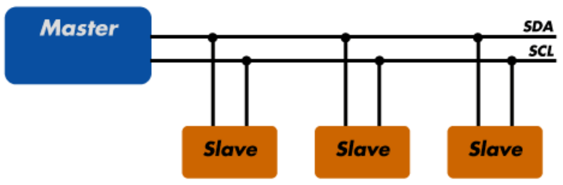

Figure2. Shows I^{2}C bus has only two lines in total

The I^{2}C bus is a two-wire serial bus, as shown in Figure 2. There's no need for chip select or arbitration logic, making it cheap and simple to implement in hardware. The two I^{2}C signals are serial data (SDA) and serial clock (SCL). Together, these signals make it possible to support serial transmission of 8-bit bytes of data-7-bit device addresses plus control bits-over the two-wire serial bus. The device that initiates a transaction on the I^{2}C bus is termed the master. The master normally controls the clock signal. A device being addressed by the master is called a slave. In a bind, an I^{2}C bus slave can hold off the master in the middle of a transaction using what's called clock stretching (the slave keeps SCL pulled low until it's ready to continue). Most I^{2}C bus slave devices don't use this feature, but every master should support it. The I^{2}C bus protocol supports multiple masters, but most system designs include only one. There may be one or more slaves on the bus. Both masters and slaves can receive and transmit data bytes. Each I^{2}C bus compatible hardware slave device comes with a predefined device address, the lower bits of which may be configurable at the board level. The master transmits the device address of the intended slave at the beginning of every transaction. Each slave is responsible for monitoring the bus and responding only to its own address. This addressing scheme limits the number of identical slave devices that can exist on an I^{2}C bus without contention, with the limit set by the number of user-configurable address bits (typically two bits, allowing up to four identical devices). In figure 3, the master begins to communicate by issuing the start condition. The master continues by sending a unique 7-bit slave device address, with the most significant bit (MSB) first. The eighth bit after the start, read/not-write, specifies whether the slave is now to receive or to transmit.

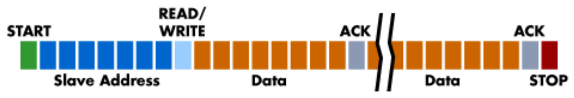

Figure3. Shows the I^{2}C bus communication

This is followed by an ACK bit issued by the receiver, acknowledging receipt of the previous byte. Then the transmitter (slave or master, as indicated by the bit) transmits a byte of data starting with the MSB. At the end of the byte, the receiver (whether master or slave) issues a new ACK bit. This 9-bit pattern is repeated if more bytes need to be transmitted. In a write transaction (slave receiving), when the master is done transmitting all of the data bytes it wants to send, it monitors the last ACK and then issues the stop condition. In a read transaction (slave transmitting), the master does not acknowledge the final byte it receives. This tells the slave that its transmission is done. The master then issues the stop condition.

IV. HARDWARE INTERFACE BETWEEN ROBOT AND I^{2}C BUS DEVICES

In this section, the hardware interfaces between I^{2}C bus system and sensors based on I^{2}C bus via serial port of PC–Card on Pioneer II, robot at institute (Fig.4). In order that the robot can receive data from sensors, the I^{2}C bus system is introduced. Because the robot has only one free serial port, it will be very hard to connect more analog or digital sensors.

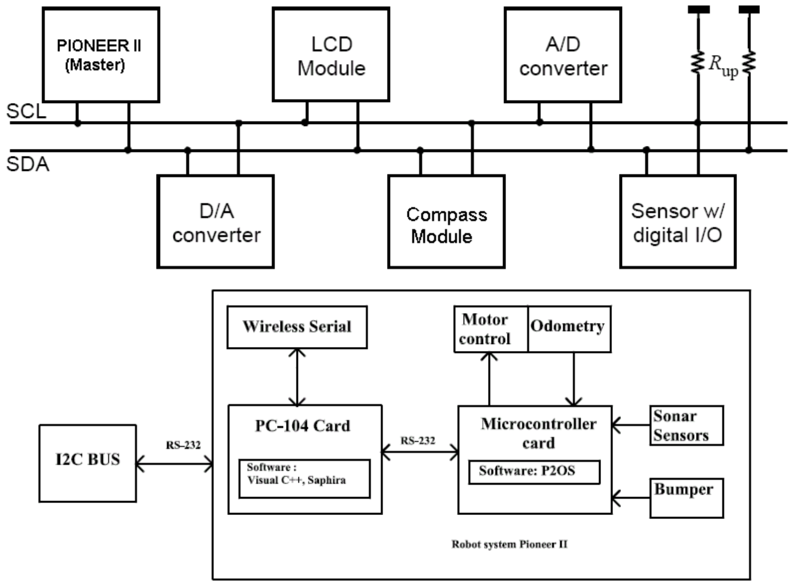

Figure4. Shows hardware interface between the robot system and I^2C bus

The previous work, microcontroller and electrical circuits are used to receive data from sensors and then send the sensor data to the robot. But it has some problems that microcontroller and electrical circuits can not be conveniently developed, if new sensors must be integrated. Because microcontroller program must be deleted and then for new sensor support reprogrammed. Furthermore, new electrical circuits must be also required for the new sensor integration. But now Philips Company has an idea to communicate between ICs (integrated circuit) and electrical module, in order that all ICs and electrical modules can be connected independently with each other. And Philips produces also more electrical modules that support I^2C bus system such as I/O port expander, analog to digital converter, RTC (time clock), LCD display and so on. For this reason I^2C bus is introduced, it is easy to connect the new sensor modules, because it can read or write data only with two lines, the serial data line (SDA) and the serial clock line (SCL). And it can connect all devices although they have different power supply (+5 Volts+9 Volts and so on). In order that the signal line of serial port from the robot can connect to devices on I^2C bus, the electrical master module must be designed to produce signal SDA and SCL. For our work, we select the electrical board from Inex-Company to produce signal and the board works as master device. This board is connected to the PC- Card in the robot through serial port. It means that the robot now works as master device to produce signal. The first I^2C bus slave device to be connected is analog to digital (ADC-Converter) converter. This module can read data from environmental analog sensors for navigation research. For our work, we have two analog sensors, a gyroscope and accelerator. This ADC-Converter module can read analog signals from sensors and then converts this analog to digital data. This device module has an important IC–PCF8591 (Fig.5) which is produced from Philips Company. The IC-PCF8591 is capable of converting four different analog voltages to the digital values and one digital value to the analog voltage. In I^2C bus system can connect eight ICs together that mean, it can support maximum 32 analog to digital inputs and four digital to analog outputs. It can also generate one analog voltage by converting an 8-bit digital value provided by the microcontroller. The next device is the I^2C slave compass sensor module (Fig.6). This sensor can work on the I^2C bus without addition circuit. This compass module has been specifically designed for use in robots as an aid to navigation. The aim was to produce a unique number to represent the direction the robot is facing.

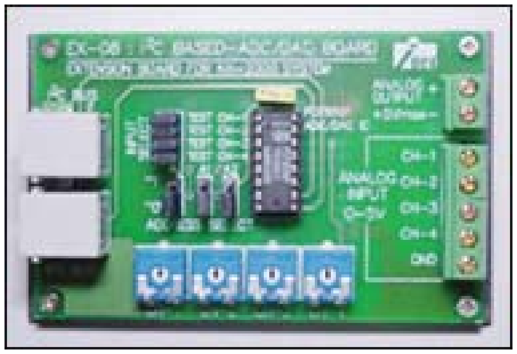

Figure5. Shows the ADC-Converter slave device

The compass uses the Philips KMZ51 magnetic field sensor, which is sensitive enough to detect the Earths magnetic field. The output from two of them mounted at right angles to each other is used to compute the direction of the horizontal component of the Earths magnetic field. The compass module requires a 5v power supply at a nominal 15mA.

The pulse width varies from 1mS (0°) to 36.99mS (359.9°) – in other words 100uS/° with a +1mS offset.

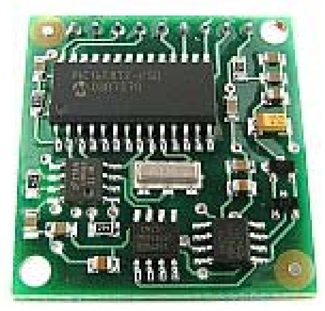

Figure6. Shows the compass module slave device

On I^2C bus, there is an important consideration that consists of the address from manufacturer and the address from user.

V. SOFTWARE DEVELOPMENT

From previous section, we describe already the communication between the Pioneer-II robot (worked as master device) and sensor module (worked as slave device). Now we will describe step by step that the robot can control SCL and SDA line to get sensor data from slave module. The software is designed to control data between the robot and ADC-Converter, Compass module, LCD module and other device, which will be added in our system in the future. This software is programmed based on Visual C^{++}. To control data line on the I^2C bus, the step ordering of function based on Visual C^{++} must be carefully accurately considered and programmed, because if one step miss or do not complete, all devices on this bus will fail. The main function used for programming will be now detailed. All conditions are only generated by robot (as master device). The two main functions are I2C_START ( ) and I2C_STOP ( ). The I2C_START ( ) function produces the START condition (Fig.7). This condition is a HIGH to LOW transition on the SDA line while SCL is HIGH. And the I2C_STOP ( ) will produce the STOP condition. This procedure is a LOW to HIGH transition on the SDA line while SCL is HIGH. Before the START condition will begin and after STOP condition finished, the bus is considered to be always free condition, the both lines must be HIGH. The next main function is I2C_ACK ( ). From figure 8, after the robot (master device) sent data to slave device finished, the slave device must send back the acknowledgement that the slave device received data already. Figure 9 shows a complete transfer cycle associated with a frame of data. First the master initiates a write by asserting logic- 0 at bit - 8, where a slave address is defined by the other 7- bits. A acknowledge signal then follows from the slave as specified in bit-9. The second and third bytes are the data and acknowledge signal. The 7-bits addressing allows 127 devices on the I^2C bus, by the use of 2-bytes address this can be extended further. The last two main functions are to control and get data from slave device. It consists of I2C_SEND ( ) and I2C_RECEIVE ( ) function. This I2C_SEND ( ) function is sent always by robot (master device) to control and set slave devices configuration and the I2C_RECEIVE ( ) function is also sent by robot to receive data from slave devices. Now we show the example function ordering to connect a slave device through I^2C bus to receive data. At first, the robot (master device) connects ADC-Converter, Compass module or LCD module that it depends on the second step, which slave device address (ADC-Converter or Compass module) will be called. The robot can be programmed to get data from ADC-Converter with function ordering. All step functions are programmed based on Visual C^{++} and detailed below

Step 1 : I2C_START ( ),
Step 2 : I2C_SEND ( ) - Call slave device address (ADC-Converter or Compass) from robot and write mode configuration,
Step 3 : I2C_ACK ( ),
Step 4 : I2C_SEND ( ) - Enable analog output, single mode and analog channel selection,
Step 5 : I2C_ACK ( ),
Step 6 : I2C_STOP ( ),
Step 7 : I2C_START ( ),
Step 8 : I2C_SEND ( ) - Read mode; start to read data from analog input of selected channel,
Step 9 : I2C_ACK ( ),
Step 10 : I2C_RECEIVE ( ),
Step 11 : I2C_ACK ( ),
Step 12 : I2C_STOP ( ),

With these steps, the robot can get data from only one module for one time. If it wants to get data from this module once again or from another module, the robot must rerun once again from step 1 to step 12.

VI. CONCLUSIONS

For this paper has mainly purpose to introduce a new way to communicate between mobile robot and sensors. The I^2C bus system is considered to use in our system, because it can be conveniently developed and modified our system, if new sensors must be integrated or more analog sensors are used. The I^2C bus system consists of master and slave devices that the master device is integrated on the Pioneer-II robot and ADC-Converter for analog sensors and compass module work as slave devices. The integrated circuits or slave devices, which support I^2C Bus System. It works only with two signal lines and has independently electrical module of each other. Considerately, the device address must be only correctly specified that the company has own configuration for each device. The I^2C bus can communicate very well via serial port of PC-Card and has no problem with robot system. For software developing, the software is developed to communicate the robot and all analog sensors by means of the I^2C bus system. The developed software is programmed in function ordering.

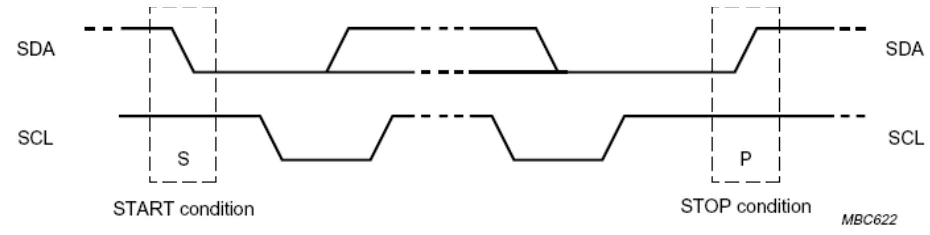

Figure7. Shows start and stop condition of two lines

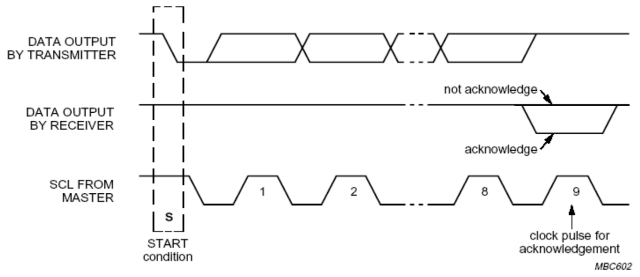

Figure8. Shows acknowledge condition

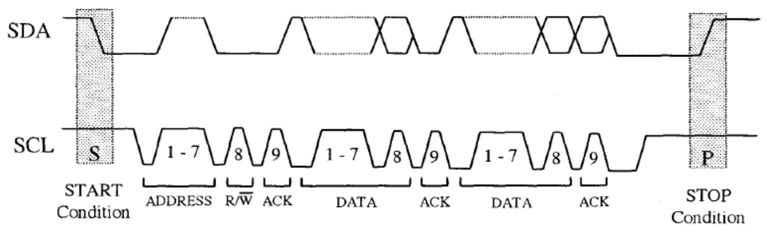

Figure9. Shows data transfer on the I^{2}C bus.

All functions must be programmed in the I^{2}C bus format to control SDA and SCL lines in order that the robot as master device can receive sensor data from slave device. This software is programmed by Visual C^{++} and combined with Saphira / Aria library from Active Media Company.

REFERENCES

[1] D. Kalinsky and R. Kalinsky, "Introduction to I^{2}C," *Embedded Systems Programming*, Aug. 2001.

[2] S. Sarns and J. Woehr. "Exploring I^{2}C," *Embedded Systems Programming*, p. 46, Sept. 1991.

[3] J. M. Flynn, "Understanding and Using the I^{2}C Bus," *Embedded Systems Programming*, Nov. 1997.

[4] Philips Semiconductor, I^{2}C Bus Specification, version 2.1, January 2000.

[5] ActivMedia (1997). *Saphira Software Manual,* ActivMedia Robotics LLC, 44 Concord Street, Peterborough, NH 03458, USA.

[6] ActivMedia (2001). *Pioneer 2 / PeopleBot Operations Manual,* 8. Aufl., ActivMedia Robotics LLC, 44 Concord Street, Peterborough, NH 03458, USA

[7] Dr. Susanne Wigard, " Visual C++ 6," Das bhv Taschenbuch.