AN EMBEDDED I²C BUS MONITORING SYSTEM DESIGN IMPLEMENTATION

Sedat ATMACA, A.Turan ÖZCERIT

Abstract - **In this article, hardware based, stand alone 100Kbit/s speed I²C bus monitor is carried out.**
I²C traffic is logged to the local on-board memory. With the help of hard and software filter the stored messages can be limited to the interesting ones.
All bus activity including start/stop events, slave addresses, read/write requests, acknowledgments, and data are displayed in computer.

Index Terms – I²C bus, I²C bus controller, serial buses, bus monitoring,

Özet – **Bu çalışmada, donanım tabanlı 100Kbit/sn hızında I²C monitor yapılmıştır.**
I²C-Bus'taki veriler, sistemdeki dahili hafızaya kaydedilerek bilgisayarın seri portundan gönderilir. Sistem kullanıcıya Bus'taki START, STOP, Yönetilen-adres, veri, oku/yaz ve kabul bilgilerinin durumu hakkında bilgi verir.

Anahtar Kelimeler– I²C bus, I²C bus denetleyicisi, serial bus'lar, bus görüntüleme.

I. INTRODUCTION

The I²C-bus is a protocol which supports the communication of the various chips in embedded systems or portable devices.
This article explains software and hardware specification for monitoring an I²C-bus with the standard 80c51 microcontroller. The I²C-bus monitor block diagram is illustrated in Figure 2. The PCF8584 is a controller which can listen to I²C-bus. This chip can be used to listen and monitor the actual data on the I²C-bus. It is illustrated in Figure 3.

Sedat Atmaca, Sakarya University, Institute of Sciences, Sakarya, sedatatmaca@hotmail.com
A.Turan Özcerit, Sakarya University, Technical Education Faculty, Sakarya, aozcerit@sakarya.edu.tr

It provides what is going on the I²C-bus. It has several internal register to tell it what to do and how to act upon the I²C-bus line. The 80c51 sends control and data bytes to control the PCF8584 in monitor mode. Figure 4 shows us I²C-bus monitor flowchart with the PCF8584.

II. THE I²C BUS CONCEPT

The I²C bus is a simple bidirectional two-wire interface that provides for efficient Inter-IC control. The function of these devices range from EEPROMs to LCD drivers.[1]
Two wires, serial data (SDA) and serial clock (SCL), carry information between the devices connected to the bus. In order to distinguish between devices on the bus, each device is recognized by a unique address (whether it's a micro-controller, LCD driver, memory or keyboard interface) and can operate as either a transmitter or receiver, depending on the function of the devices. Obviously an LCD driver is only a receiver, whereas a memory can both receive and transmit data. In additon to transmitters and receivers, devices can also be considered as masters or slaves when performing data transfers.
The I²C-bus is a multi-master bus. This means that more than one devices capable of controlling the bus can be connected to it. As masters are usually micro-controllers, let's consider the case of a data transfer between two micro-controllers connected to the bus.[2]
Both SDA and SCL are bi-directional lines. When the bus is free, both lines are high. Devices are capable of clamping and releasing the wires. If one of the devices connected to the bus clamps a line, this line will become low and only if devices have released a line, it will become high again.
The I²C-bus protocol uses two wires for communication. One is the data line (SDA) and one is the synchronization line (SCL). The data is transmitted in packets of 8 bits (one bytes), followed by an acknowledgement bit from the receiver. For the entire message a few extra bits are added to indicate start

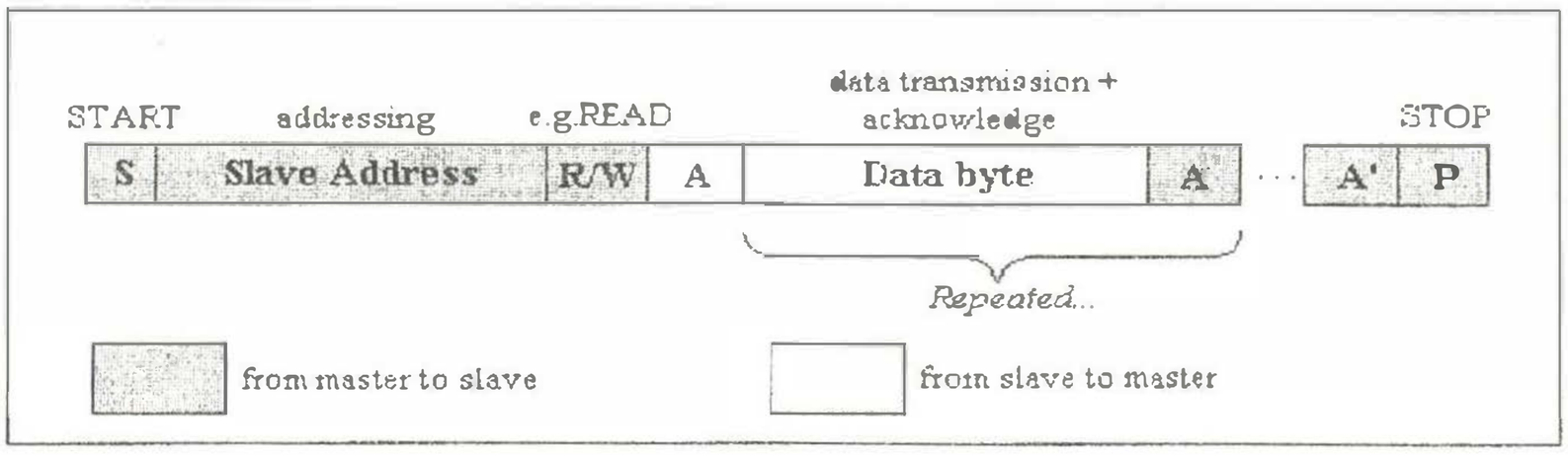

Figure 1. I²C bus protocol

conditions and stop conditions. Those conditions tell other devices whether the bus is currently in use or not. The protocol also provides a simple but effective means to handle data collision. Data collision is a state where in data is blurred because more devices try to transmit bytes at the same time.

In Figure 1 a possible message is illustrated. The message is initialized by a device acting as a master. The device one meant to respond acts as a slave. The message begins with a START condition followed by the address of the slave devices.

If anything goes wrong, the receiving device is busy doing something else, it can always respond with no acknowledgement. The master stops the transmission and may try later.[3]

connected to the 8-bit data bus of a microprocessor, together with the bus control lines. The interface and connections for the PCF8584 are shown in Figure 3 in detail.

This IC is a powerful and versatile means of providing I²C communications for an embedded system.

This chip has a certain mode in which it does not take part in the real I²C communication but only records what is going on. It listens to all addresses, but does not generate any acknowledge.

A universal I²C data logger system has been built using some software routines and a MCU.

The PCF8584 is used to interface between parallel microprocessor or microcontroller buses and the serial I²C-bus. On the I²C-bus, it can act either as a master or a slave. Bi-directional data transfer between the I²C-bus

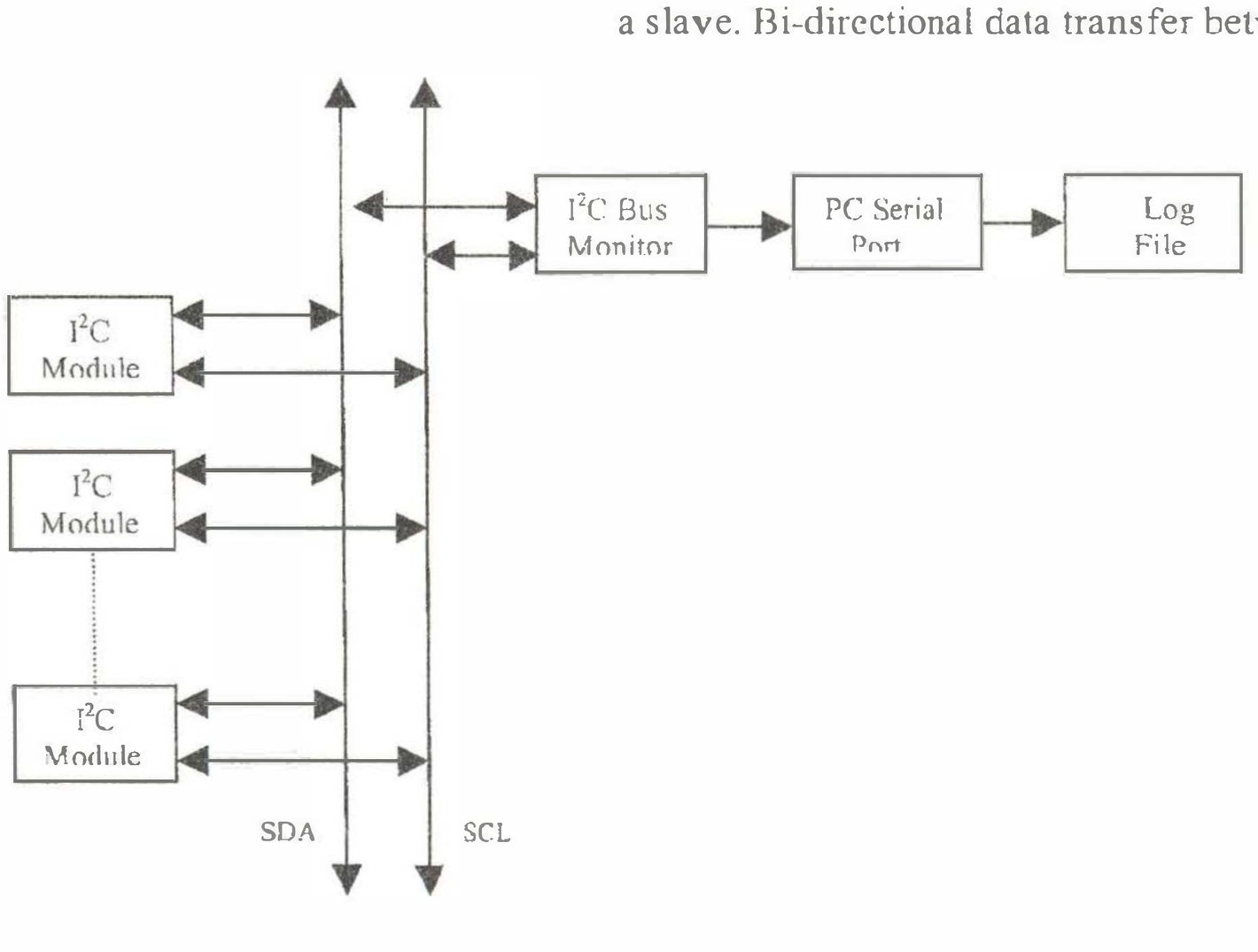

Figure 2. I2C Bus connection diagram

III. I²C BUS MONITORING SYSTEM

The I²C bus monitor system block diagram is illustrated in Figure 2. It can be designed in several ways. One of them is software oriented, another is hardware oriented. In this study, I²C bus is monitored with hardware oriented I²C bus controller namely PCF8584. The PCF8584 is a device that can be

and the parallel-bus of a microcontroller is carried out on a byte-wise basis, using either an interrupt or polled handshake.[4]

The PCF8584 has five internal register locations. Three of these (own address register S0', clock register S2 and interrupt vector S3) are used for initialization of the PCF8584. Normally they are only written once directly after resetting of the PCF8584.
In order to control PCF8584, the 80c51 has been used with an 8-bit control register and 8-bit status register. Those registers are located on the PCF8584 and can be accessed via the same data lines that it is used to transfer data. To choose between data to transceive and data to control the chip, a hardware line is required. This line is called A0.
PCF8584 status/control register is called S1. This register 8-bit wide and 2-level in depth. The first level consists of eight write-only bits and it is the control section. The second level consists of eight read-only bits and it is the status section.

The most significant bit in the S1 register is PIN (Pending Interrupt Not). This bit can be read as well as written. Bus traffic is monitored by the PIN bit, which is reset to logic 0 after the acknowledge bit of an incoming byte has been received, and is set to logic 1 as soon as the fist bit of the next incoming byte is detected. Reading the data buffer S0 sets the PIN bit to logic 1. Data in the read buffer is valid from PIN = 0 and during the next 8 clock pulses (until next acknowledge).
The second bit is ESO (Enable serial Output) and can be used to switch the serial I^2C-interface on or off. When the interface is shut off a few special control data can be programmed into the chip. Those data are its own address (register S0'), an interrupt vector (register 3) and a bit sequence to select internal and I^2C clock rates (register 2). When 7-bit own address register S0' is loaded with all zeros, the I^2C controller acts as a passive I^2C monitor. To select one of those special registers the ES1 and ES2 bits should be programmed according to specification. An external interrupt output can be enabled with ENI and the generation of S*tart* and S*top* conditions for serial communications can be controlled by the STA and STO bits. With ACK the sending of acknowledges after each transmitted byte can be controlled.
he following bits can only be read and only used in systems monitor:

- In monitor mode the controller is always in Slave/Receiver mode.
- The controller never generates an acknowledge.
- The controller never generates an interrupt request.
- A pending interrupt condition does not force SCL LOW.
- Received data is automatically transferred to the read buffer.[5]

The STS bit tells us if someone generates a *Stop* condition on the I^2C bus. If BER becomes logic true, a bus error has been detected. Automatically BB' is reset to 1 (inactive) and PIN is set to 0 (active).
The Another status bit is LRB/AD0, 'Last Received Bit' or 'Address 0 (General Call) bit'. This status bit serves a dual function, and is valid only while PIN = 0; First, LRB holds the value of the last received bit over the I^2C-bus while AAS = 0 (not addressed as slave). Normally this will be the value of the slave acknowledgement; thus checking for slave acknowledgement is done via testing or the LRB.
Second, AD0; when AAS = 1 ('Addressed As Slave' condition), the I^2C-bus controller has been addressed as a slave. Under this condition, this bit becomes the 'AD0' bit and will be set to logic 1 if the slave address received was the 'general call' (00H) address, or logic 0 if it was the I^2C-bus controller's own slave address.

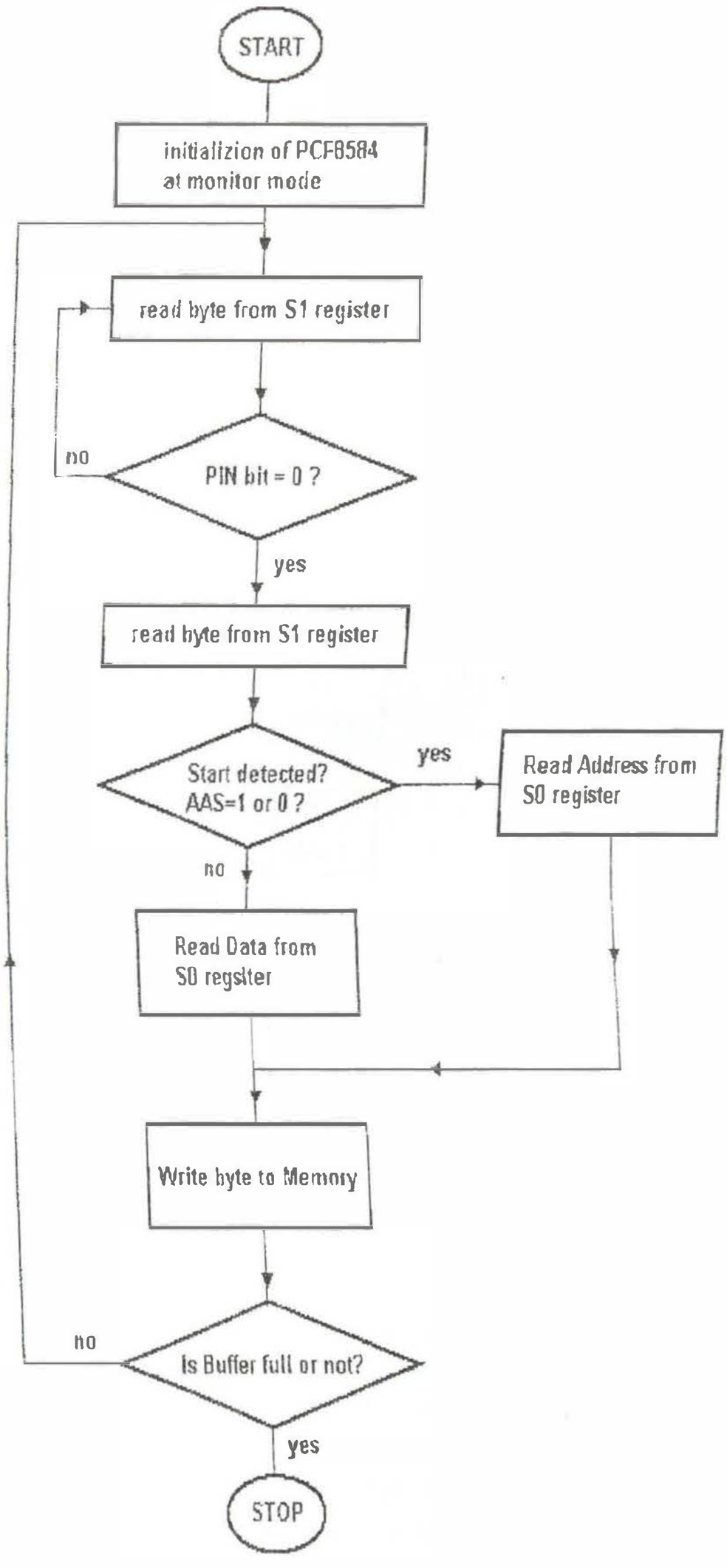

Figure 4. I^2C Bus monitor

Next is the AAS bit (Addressed As Slave). The bit becomes active when the address signaled from the bus matches our chip's address. AAS is set to logic 1 at

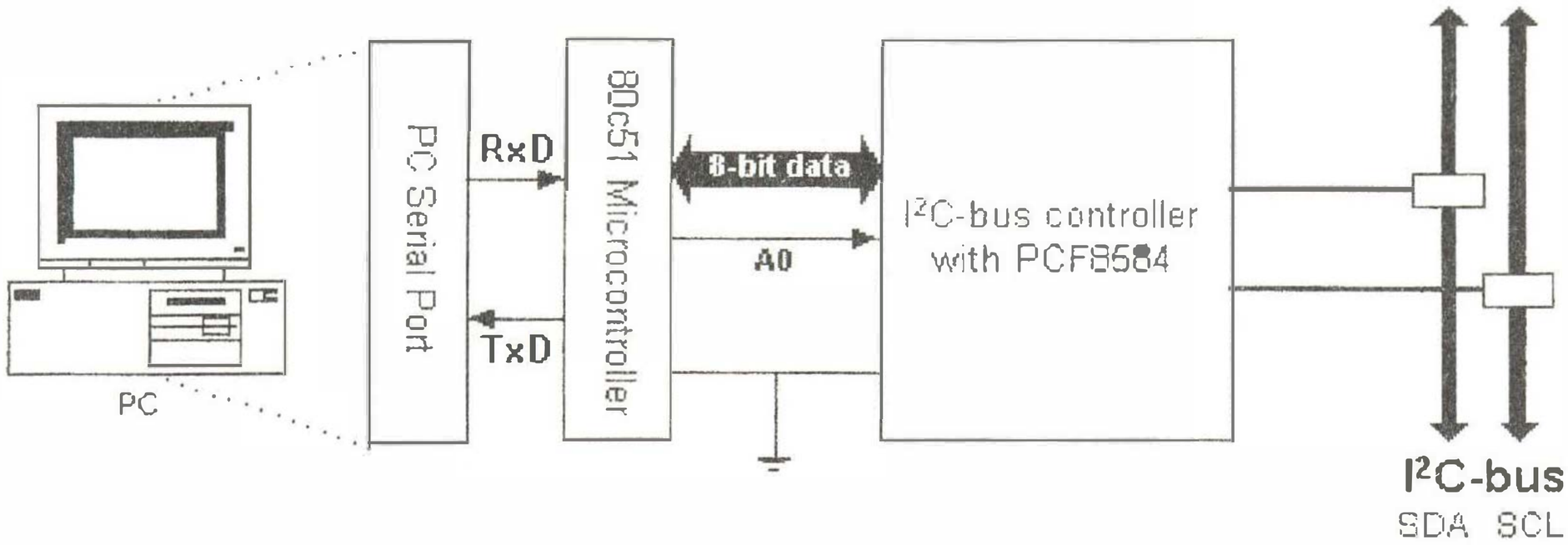

Figure 3. The I²C Monitor with PCF8584

every *Start* condition, and reset at every 9th clock pulse. The LAB bit (Lost Arbitration) is set when another device has taken over master. If so, we loose our control and become slave. Last bit in the status register is BB'(Bus Busy, reverse logic). If 0 the bus is currently in use and access can better be postponed.
But if we want to be rascals we could just transmit some data bits- to tease the other devices that try to communicate.

IV.CONCLUSIONS

In this study I²C-bus monitor system listens and monitors data with the help of PCF8584 hardware based I²C-bus controller, and the 80c51 microcontroller. It can be used for testing available signals on the I²C-bus and error conditions. Because of this, it can be used as test equipment in a digital laboratory. I²C-bus actions are logged to the on-board memory and the system designed filters the stored messages that are limited to the interesting ones.
I²C-bus monitor is able to operate at a 100Khz clock speed. Because the average execution period for an 80c51 instruction is 12 cycles, the microcontroller using a 11.0592MHz clock can expect to average 1μS per instruction. It appeared, that there would be about 10 instructions worth of execution time during a normal I²C clock period. In order to monitor upon the 100KHz bus devices (400Khz), It would be necessary to employ one of the faster 8051s that run at 40Mhz clock speeds.

REFERENCES

[1] James, M.Flynn, "Understanding and Using the I²C-Bus", Embedded System Programming
[2] Philips Semiconductors, "The I²C-Bus Specification", Version 2.1, 2000
[3] Koetsier Hilbert, "Personal Computer interface to I2C bus via parallel printer port using PCF8584 bus controller, System basics and Specification, 1999
[4] Philips Semiconductors, "Interfacing the PCF8584 I2C-Bus controller to 80c51 family microcontrollers", 1994
[5] Philips Semiconductors, "PCF8584 I²C-Bus Controller", 1997