# *Real Time Embedded PID Controller using ARM Controller and MODBUS Protocol*

Alpesh Patel[1], Naimesh Mehta[2]

*1. M. E- II year, Instrumentation and Control Engineering, L. D. College Of Engineering Ahmadabad.*

*2. Faculty of Instrumentation and Control Engineering Department, Ahmadabad.*

*1alpeshpatel_ic@yahoo.com 2 naimesh1968@yahoo.com*

**Abstract—embedded systems are used in all aspects of modern life and examples of their use are numerous. Therefore more and more embedded system developers and system-on-chip designers' rely on microprocessor based design methodology to reduce time-to market and high performance. One of the most important hardware components of an embedded system is microprocessor/microcontroller. The embedded ARM system can be used to full-fill strict requirements of the data acquisition system, such as the function, reliability, cost, size, power consumption etc. ARM Controllers are sophisticated multiprocessor design which has RISC type architectures. ARM variants have not only their high performance and low cost advantages but also power saving features. In the industries there are no of process loops (SISO and MIMO) necessitate tuning. More than 90% of all control loops are PID. The proportional –integral –derivative (PID) controllers are designed for application where load change are expected and need control action fast and accurately. In the present paper ARM processor is used to implement Proportional-Derivative-Integral controller card for maintaining a specific level in the level control trainer. Modbus communication would be implemented between ARM processor and personal computer through serial port with help of labview.**

***Keywords-ARM comtroller;MODBUS;PID controller; serial communication;labview;***

## I. Introduction

Embedded system is a special-purpose computer system designed to perform a small set of dedicated functions, sometimes with real-time computing constraints. Some of the known applications are consumer electronics including personal digital assistants (PDAs), mobile network system, anti-lock breaking system (ABS), GPS, electronic stethoscope.

A series of ARM processors have RISC type architectures, and they have been widely used in a number of embedded designs ARM Controllers are sophisticated multiprocessor design using the fastest processors on the market today. It is important to understand how the hardware works and the restraints that using a certain peripheral may have on the rest of the system. It is essential to know how to develop the software for such systems and the effect that different hardware designs can have on the software and vice versa [1].

In industrial process control loops, proportional-integral-derivative is extensively used to optimize process desired control action. PID controller is still used widely in presence of fuzzy, adaptive and many more advance control techniques. More than 90% of all control loops are PID [4]. PID controller is like a heart of feedback loop because of simple and effective structure. The PID algorithm is simple and used in many feedback loops. Different forms of PID algorithm like interacting and non-interacting, set-point weighting, standard form, classical form, parallel form are used as per control action demand [4].

The MODBUS protocol provides the internal standard that the Modicon controllers use for parsing massages for serial communication [3].

## II. MODBUS Protocol

The MODBUS protocol provides the internal standard that the Modicon controllers use for parsing massages. During communications, the protocol determines how each controller will know its device address, recognize a massage addressed to it, determine the kind of action to be taken, and extract any data or other information contained in the massage. If a reply is required, the controller will construct the reply massage and send it using MODBUS protocol. In massage transmissions, the MODBUS protocol embedded into each networks packet structure provides the common language by which the device can change data [2].

Standard MODBUS ports on Modicon controllers use an RS-232C compatible serial interface that defines connector pin-outs, cabling, signal levels, transmission baud rates, and parity checking. Controllers can be networked directly or via modems. Controllers communicate using a master-slave technique, in which only one device (the master) can initiate transactions (queries) [2].The other devices (the slaves) respond by supplying the requested data to the master, or by taking the action requested in the query. Typical master devices include host processors and programming panels. Typical slaves include programmable controllers.

The master can address individual slaves, or can initiate a broadcast message to all slaves. Slaves return a message (response) to queries that are addressed to them individually. Responses are not returned to broadcast queries from the master. The MODBUS protocol establishes the format for the master's query by placing into it the device (or broadcast) address, a function code defining the requested action, any data to be sent, and an error-checking field. The slave's response message is also constructed using MODBUS protocol. It contains fields confirming the action taken, any data to be returned, and an error-checking field. If an error occurred in receipt of the message, or if the slave is unable to perform the requested action, the slave will construct an error message and send it as its response.

In RTU mode, messages start with a silent interval of at least 3.5 character times [3]. This is most easily implemented as a multiple of character times at the baud rate that is being used on the network (shown as T1-T2-T3-T4 in the figure below). The first field then transmitted is the device address. The allowable characters transmitted for all fields are hexadecimal 0 through 9, and then A through F. Networked devices monitor the network bus continuously, including during the silent intervals. When the first field (the address field) is received, each device decodes it to find out if it is the addressed device. Following the last transmitted character, a similar interval of at least 3.5 character times marks the end of the message. A new message can begin after this interval. The entire message frame must be transmitted as a continuous stream. If a silent interval of more than 1.5 character times occurs before completion of the frame, the receiving device flushes the incomplete message and assumes that the next byte will be the address field of a new message. Similarly if a new message begins earlier than 3.5 character times following a previous message, the receiving device will consider it a continuation of the previous message. This will set an error, as the value in the final CRC field will not be valid for the combined messages.

| START | ADDRESS | FUNCTION | DATA | CRC CHECK | END |
|---|---|---|---|---|---|
| 1 CHAR | 2 CHAR | 2 CHAR | n CHAR | 2 CHAR | 2 CHAR CRLF |

Fig.1 Modbus frame

## III. ARM CONTROLLER HISTORY

The ARM is a 32-bit reduced instruction set computer (RISC) instruction set architecture (ISA) developed by ARM Holdings. It was known as the Advanced RISC Machine, and before that as the Acorn RISC Machine. The ARM architecture is the most widely used 32-bit ISA in terms of numbers produced [1]. They were originally conceived as a processor for desktop personal computers by Acorn Computers, a market now dominated by the x 86 families used by IBM PC compatible and Apple Macintosh computers.

The LPC2101/2102/2103 microcontrollers are based on a 16-bit/32-bit ARM7TDMI-S CPU with real-time emulation that combines the microcontroller with 8 kB, 16 kB or 32 kB of embedded high-speed flash memory. A 128-bit wide memory interface and unique accelerator architecture enable 32-bit code execution at the maximum clock rate. For critical performance in interrupt service routines and DSP algorithms, this increases performance up to 30 % over Thumb mode. For critical code size applications, the alternative 16-bit Thumb mode reduces code by more than 30 % with minimal performance penalty. Due to their tiny size and low power consumption, the LPC2101/2102/2103 is ideal for applications where miniaturization is a key requirement.

A blend of serial communications interfaces ranging from multiple UARTs, SPI to SSP and two I2C-buses, combined with on-chip SRAM of 2 kB/4 kB/8 kB, make these devices very well suited for communication gateways and protocol converters. The superior performance also makes these devices suitable for use as math coprocessors. Various 32-bit and 16-bit timers, an improved 10-bit ADC, PWM features through output match on all timers, and 32 fast GPIO lines with up to nine edge or level sensitive external interrupt pins make these microcontrollers particularly suitable for industrial control and medical systems.

## IV. PID CONTROLLER

As conventional controllers we can count a controllers known for years now, such as P, PI, PD, PID, Otto-Smith, all their different types and realizations, and other controller types. It is a characteristic of all conventional controllers that one has to know a mathematical model of the process in order to design a controller. Unconventional controllers utilize a new approaches to the controller design in which knowledge of a mathematical model of a process generally is not required. Examples of unconventional controller are a fuzzy controller and neuro or neuro-fuzzy controllers.

Manny industrial processes are nonlinear and thus complicate to describe mathematically. However, it is known that a good many nonlinear processes can satisfactory controlled using PID controllers providing that controller parameters are tuned well. Practical experience shows that this type of control has a lot of sense since its simple and based on 3 basic behavior types: proportional (P), integrative (I) and derivative (D) [4].

Instead of using a small number of complex controllers, a larger number of simple PID controllers are used to control simpler processes in an industrial assembly in order to automate the certain more complex process. PID controller and its different types such as P, PI and PD controllers are today basic building blocks in control of various processes.

In spite their simplicity; they can be used to solve even a very complex control problems, especially when combined with different functional blocks, filters (compensators or correction blocks), selectors etc. A continuous development of new control algorithms insure that the time of PID controller has not past and that this basic algorithm will have its part to play in process control in foreseeable future. It can be expected that it will be a backbone of many complex control systems [4].

There are a number of different PID controller structures. Different manufacturers design controllers in different manner. However, two topologies are the most often case:
· Parallel (non-interactive)
· Serial (interactive)

## V. IMPLEMENTION OF PID CONTROLLER FOR LEVEL TRAINER

The level control setup used in the present project work is shown in Fig 3. As shown in fig.4 PID implementation based on ARM, a specific level set point is given through user interface and according to measured value of current level the controller output is in such a way that level is maintained at set value. In order to maintain the specific level in the tank ARM controller with PID controls and Modbus operating system accurate control would be implemented. The sequence of control action is shown in block diagram fig.2. The capacitance probe type level transmitter is used for the measurement of the tank level. The output signal from the transmitter is in 4t020ma range as per the calibrated 0-100% value of tank level. Then the through ItoV convertor the actual signal or value of level is goes into PID controller .The level set point given through Graphical User Interface (GUI) through LABVIEW to the controller. And the error is calculated in controller and corrective control/output signal calculated and final control element control valve is operated with help of VtoI and ItoP convertor and desired level is controlled. The hardware incorporates tank, control valve, level transmitter, I-P convertor, DAC, ARM controller etc as shown in Fig.4 and fig.5.

PID controller software is implemented in KEIL tool for ARM and the flowchart for the program is shown in fig.7. The Modbus RTU is used for the serial communication. The PC (GUI in Labview) act as the master and PID controller as the slave. Experiment result is shown in fig 8.

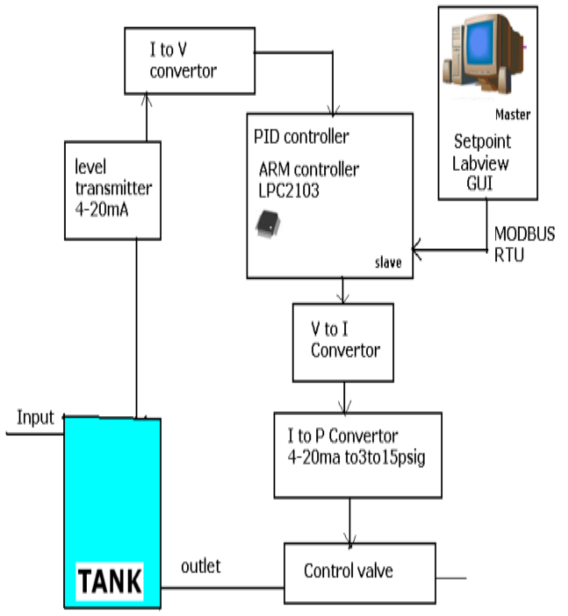


Fig.2 Block diagram of system

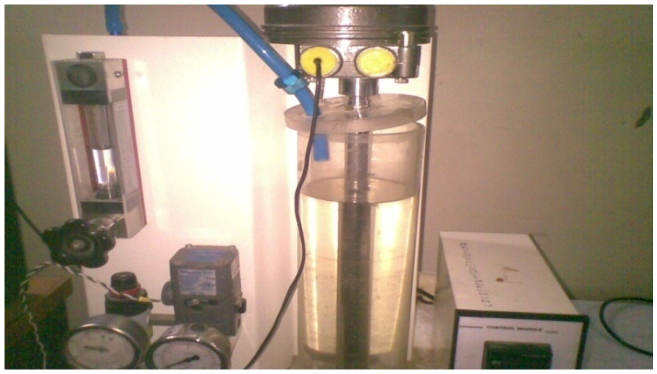


Fig.3 Level Control trainer

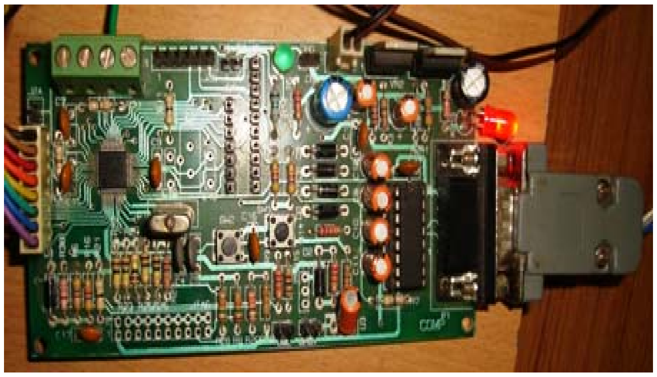


Fig.4 PID Controller using ARM controller

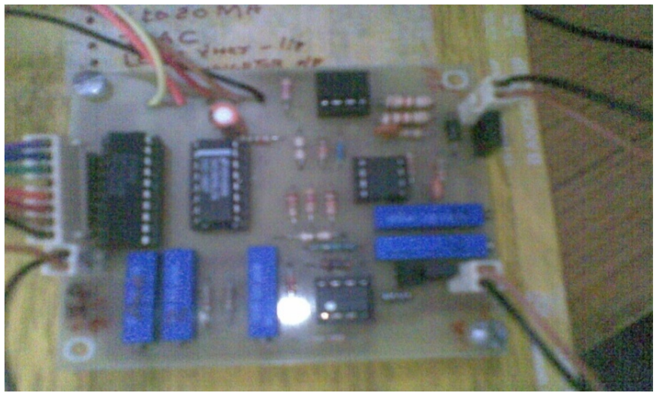
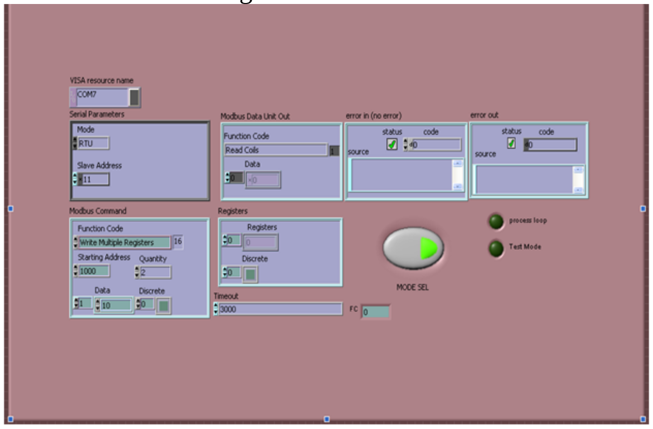


Fig.5 ItoV and DAC



VISA resource name
COM7
Serial Parameters
Mode RTU
Slave Address 11
Modbus Data Unit Out
Function Code Read Coils 1
Data 0 0
error in (no error)
status ✓ code 0
source
error out
status ✓ code 0
source
Modbus Command
Function Code Write Multiple Registers 16
Starting Address 1000 Quantity 2
Data 1 10 Discrete 0
Registers
Registers 0 0
Discrete 0
Timeout 3000
FC 0
MODE SEL
process loop
Test Mode

Fig.6 Front panel for MODBUS window

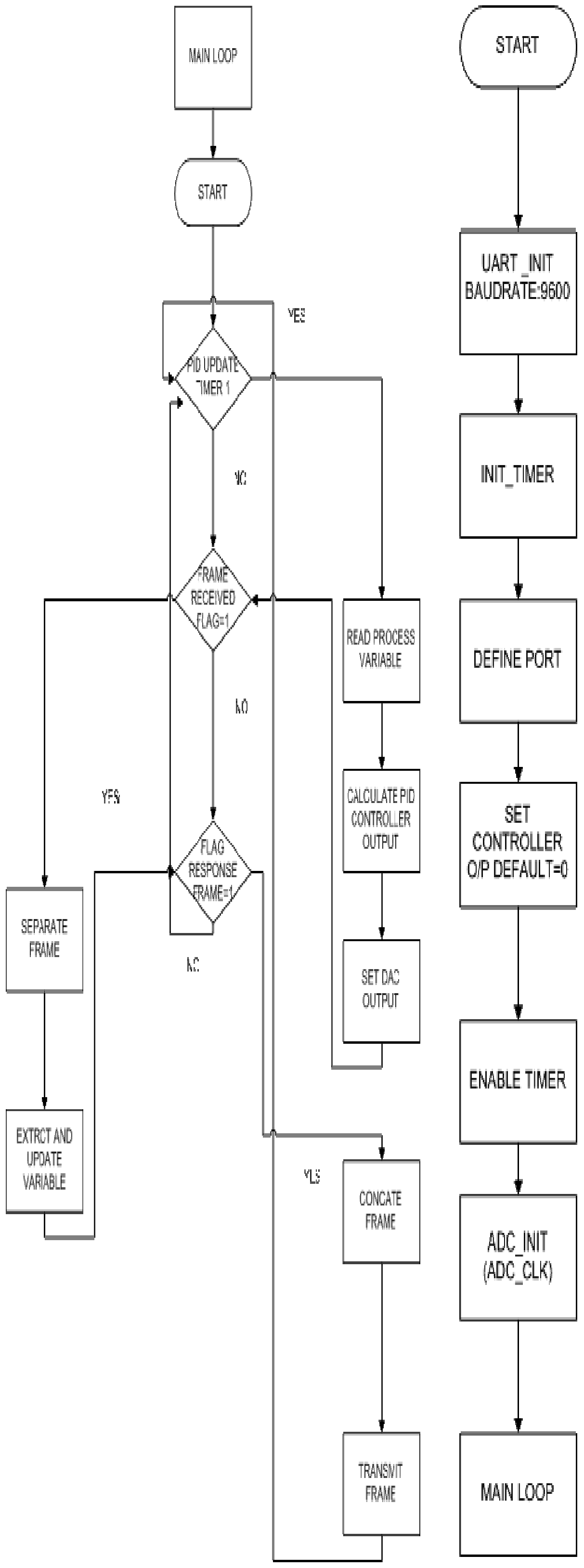


Fig.7 Flowchart of PID algorithm

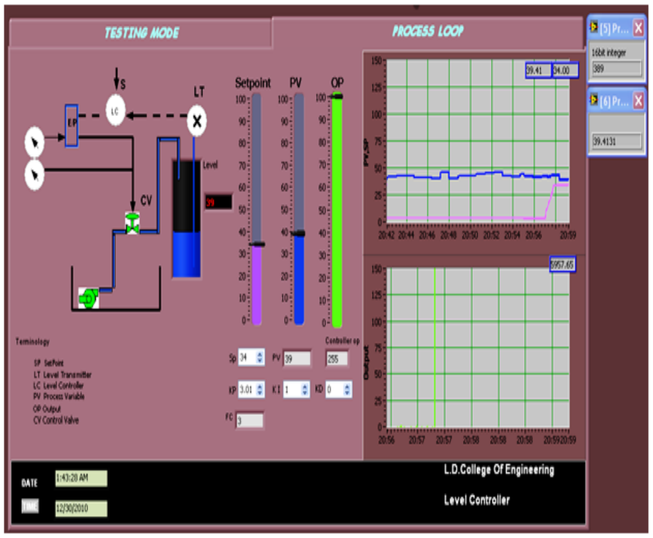


Fig.8 Level control with result

## VI. Conclusion

On the developed system, PID algorithm is successfully implemented and tested with RTU mode using Modbus protocol. Process mimic diagram developed in LABVIEW which gives not only the detail process information but also shown analog varying quantities of all parameter and trend chart also. Developed Control system is flexible, simpler and cost effective. In the presence of this system the communication function is strong, structure is simple and usage is convenient. Due to the generic design the device also can be reused in many different industrial applications with the same features like cascade control and ratio control etc.

## VII. References

[1] J. Goodacre and A. N. Sloss, "Parallelism and the ARM Instruction Set Architecture", IEEE publications, July 2005.

[2] M.popa, M Marcu and a.s.popa "a microcontroller based Data acquisition system with usb interface", university Polytechnic timisioarm, Romania.

[3] "www.modbus-ida.org".

[4] K.J. Astrom and T. haglund , "Auto Tuning of PID controllers," 1st ed. Reserch and triangle park, NC:instrum. Soc. Amer.,1988.