www.ijsrnsc.org

Available online at www.ijsrnsc.org

IJSRNSC

Volume-5, Issue-1, April- 2017
Review Paper

Int. J. Sc. Res. in Network Security and Communication
ISSN: 2321-3256

Design and Development of Reconfigurable Multiprocessor Architecture for Embedded Systems

SM. Kalyankar[1*], SD. Sawarkar[2]

[1]Dept. of Electronics and Telecommunication Department, DMCE, Airoli, India
[2]Dept. of Electronics and Telecommunication Department, DMCE, Airoli, India
Corresponding Author: shubhangikatke@gmail.com

Received 18th Jan 2017, Revised 04th Feb 2017, Accepted 12th Mar 2017, Online 30th Apr 2017

Abstract- In Today's world, Embedded systems are the brains of almost all digital and industrial control systems. Modern digital systems demand increasing electronic resources, so the multiprocessor platforms are a suitable solution for them which provides better results in terms of area, speed, and power consumption compared to traditional uniprocessor digital systems. Reconfigurable multiprocessor systems are a particular type of embedded system, implemented using reconfigurable hardware. In this paper, Design methods and challenges are discussed. Advances in FPGA technology are leading to more powerful systems in terms of processing and flexibility. Flexibility is one of the strong points of this kind of system, and multiprocessor systems can even be reconfigured at run time, allowing hardware to be adjusted to the demands of the application. Multiprocessor Systems-on-Chip (MPSoC) represent an important trend in digital embedded electronic systems. Although hardware support for parallel computationis increasingly available in embedded processing platforms, there is a distinct lack of effective software support. One of the most important issues with regards to such systems is communication between processors. Now communication in different controllers can be done mainly by two ways i.e. by using I2C and SPI protocol. But there are some limitations of these two protocols which are discussed in this paper and how it can be overcome by using our proposed protocol. Here, in this project, a new method for communication is proposed for an embedded system having multiple peripherals on the board. The adapted arrangement is parallel and hence is more faster way to communicate. This Paper will provide in-depth description of the Reconfigurable Distributed Computing Arrangement of On-board Multiprocessor Communication for Embedded Systems and will investigate its merits & demerits.

Keywords: Embedded Systems, I2C,SPI, FPGA, MPSoC, Multiprocessor Communication

I. INTRODUCTION

Now days, embedded systems are everywhere. They are becoming more and more complex to simplify our lifestyle. Many features are being embedded in a single tiny system to fulfill our everyday requirements. Thus more & more peripheral needs to be integrated in the single embedded system. This increases the design time of the system and the processing load is increased for a single microprocessor. This arises the need of such an arrangement of peripherals which can be interfaced in a network instead of directly interfacing with the processing unit. This type of arrangement of peripherals will create a distributed interface of multiple peripherals.

A. What is an Embedded System and Multiprocessing?

An embedded system is an electronic/electro-mechanical system designed to perform a specific function and is a combination of both hardware and firmware (software). They are the devices used to control, monitor or assist the operation of equipment, machinery or plant. Multi-processing refers to the ability of a system to support more than one processor at the same time and/or the ability to allocate tasks between them. Applications in a multi-processing system are broken to smaller routines that run independently.

B. What is communication Protocol?

It is the set of specific standard and rules used to form a communication between its components and peripherals so that the system works efficiently.

In this paper, we will be discussing traditional protocols like I2C and SPI protocol as well as our proposed method of communication between on board processors.

II.BACKGROUND ANALYSIS

In almost all automatic electronics devices microcontroller is decision making element. To get our work done we interface a large number of peripheral devices such as LCD, Relays, Keypad, ADC, RTC, Motors, Thermal Printers, External Memory, seven segment displays and many more to fulfill our requirement. At our level, we usually work with all ICs form 8051 family. We know these ICs are 40 pin microcontrollers with only 4 port for external world. But for our complex requirement we need to interface a large number of peripheral devices which is not possible with a single microcontroller. The limitation of using single Controller are limited code memory, only few peripheral

© 2017, IJSRNSC All Rights Reserved

devices can be interfaced, Design complexity, error probability and software complexity.

A. *Currently Available System*

The solution of above Problem is to use more than one controllers or multi controllers in a single project which leads to the communication between on board controllers. Now communication in different controllers can be done mainly by two ways i.e. I2C protocol andSPI protocol. Both protocols are well-suited for communications between integrated circuits, for slow communication with on-board peripherals.

1) SPI- Serial Peripheral Interface:SPI is asynchronous data bus commonly used to send data between microcontrollers and small peripherals such as shift registers, sensors, and SD cards. It uses separate clock and data lines, along with a select line to choose intended device.SPI Features:

- SPI does not define any maximum data rate, not any particular addressing scheme.
- It does not have an acknowledgement mechanism.
- Actually, the SPI master has no knowledge of whether a slave exists, unless 'something' additional is done outside the SPI protocol.
- SPI does not care about the physical interface characteristics like the I/O voltages and standard used between the devices.

2) I2C- Inter Integrated Communication: I2C protocol requires only 2 wires for connecting all the peripheral to a microcontroller. The original specification defined a bus speed of 100 kbps. The specification was reviewed several times, notably introducing the 400 kbps speed in 1995 and – since 1998, 3.4 Mbps for even faster peripherals. I2C Features:

- Simple bi-directional 2-wire bus, serial data (SDA)–serial clock (SCL).
- Has become a worldwide industry standard and used by all major IC manufacturers, Multi-master capable bus with arbitration feature.
- Master-Slave communication; Two-device only communication.
- Each IC on the bus is identified by its own address code.
- There is exits synchronous communication where clock frequency is maintained by the master.

B. *Limitations of Existing System*

- Synchronization of the clock is required.
- Data is transferred serially hence the system becomes slow.
- Master slave configuration is strictly maintained.
- Slaves cannot transfer data directly among each other
- All data transfer has to take place through master itself.

C. *Proposed System*

- We use 8-bit parallel data lines to transfer the data and 3-bit parallel line for handshaking signal.
- There will be no master Slave configuration.
- Each controller can communicate or transfer data independently.
- Priorities can be set by the controller's IDs.
- We developed an algorithm by which Up to 255 microcontrollers can transfer their data to desire destination microcontroller independently.
- Data transfer here means that any microcontroller can get data from any controller but with its permission. Similarly any microcontroller can send data from itself to any destination controller.
- The clock frequencies of these controllers can may or may not be same but the protocol will work efficiently.
- The same program will be burn in all the controllers without any modification.
- Addition and removal of any controller will not affect the protocol.

III. SYSTEMOVERVIEW

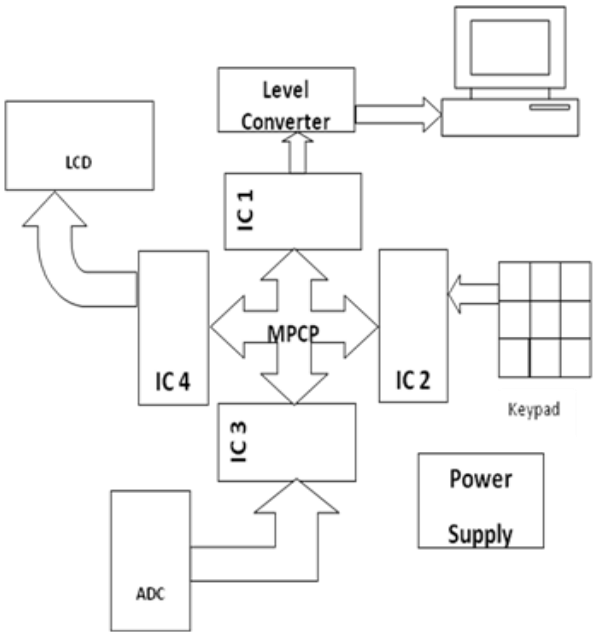

Fig.1 Block Diagram of Reconfigurable Multiprocessor Architecture

A. *The Microcontroller*

This unit is the heart of the complete system. It is actually responsible for all the process being executed. An embedded microcontroller is a chip which has a computer processor with all its support functions (clock & reset), memory (both program and data), and I/O (including bus interface) built into the device. These built-in functions minimize the need for external circuits and devices to be designed in the final application. The job of this is to continuously scan the data from PC, to display the messages on the LCD, to communicate with the SD card etc. In short, we can say that the complete intelligence of the project resides in the software code embedded in the microcontroller. In this protocol, we are using AT89C51 microcontroller.

B. *PC Interface Unit:*

© 2017, IJSRNSC All Rights Reserved

This unit will provide Interfacing between the Microcontroller which itself works on TTL Logic to the Computer which works on RS232 Logic.

C. ADC :

An analog-to-digital converter is an electronic circuit that converts continuous signals to discrete digital numbers. The digital output may be using different coding schemes, such as binary and two's complement binary. Analog-to-digital (A/D) converters are used to transform analog information, such as audio signals or measurements of physical variables (for example, temperature, force, or shaft rotation) into a form suitable for digital handling, which might involve any of these operations: (1) processing by a computer or by logic circuits, including arithmetical operations, comparison, sorting, ordering, and code conversion, (2) storage until ready for further handling, (3) display in numerical or graphical form, and (4) transmission.

D. Display Unit

This unit is used to display all the system messages. Various display device such as seven segment display. LCD display, etc can be interfaced with microcontroller to read the output directly. In our protocol, we use a two line LCD display with 16 characters each. LCD displays temperature of the measured element, which is calculated by the microcontroller. CMOS technology makes the device ideal for application in hand held, portable and other battery instruction with low power consumption.

E. Keypad

This unit is used by the user to enter the required data. Keypad is basically a matrix of many rows and columns. As we press a particular key on the keypad, the spacing between two wires reduces until the two wires, of one particular row and one particular column, touch each other. Different scanning techniques may be developed to determine which key is pressed (termed as keypad scanning).

F. Power Supply Unit

This unit will supply the various voltage requirements of each unit. This unit will be consisting of transformer, rectifier, filter and regulator. For our protocol, we require +5V regulated power supply with maximum current rating 500Ma.

IV.SYSTEM ARCHITECTURE

Architecture of our proposed system is shown in fig.2. There are dedicated microcontrollers attached to each individual peripheral or I/O devices viz. LCD, ADC, Keypad, ALU, etc.

Up to 255 I/O devices can be attached with this configuration.

A. Hardware Interface

There will be 8-bit parallel Data lines that are connected to all controllers; we will call it the 'Bus'.

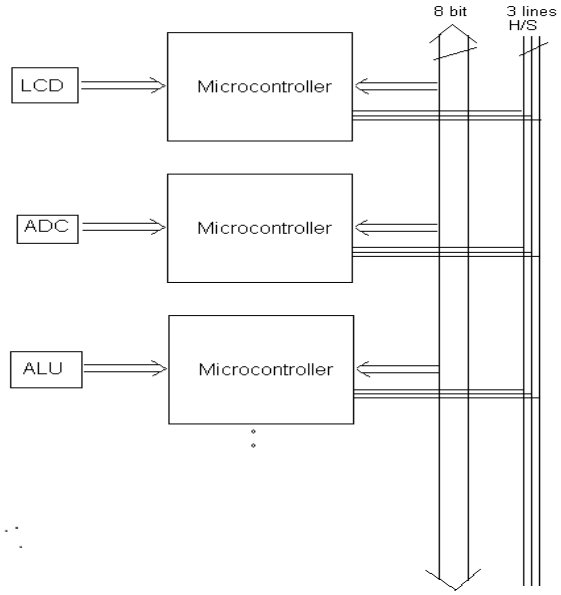

Fig. 2 Architectural view of multiprocessor communication protocol

There will be 3-bit handshaking or control lines which are defined below.

First signal line (CS) indicates the status of the 8-bit Data bus

If CS = 0 bus is busy
If CS = 1 bus is free

Second signal line (SACK) sender's acknowledgment

SACK = 1 IDLE
SACK = 0 ACTIVE

Third signal line (RACK) received acknowledgment

SACK = 1 IDLE
SACK = 0 ACTIVE

Hence there will be 11 line only, for providing communication between up to 255 microcontrollers.

B. Developing the Protocol

Step 1: Define packet format.
Step 2: Defining Algorithm.
Step 3: Connection establishment.
Step 4: Packet transfer.
Step 5: time slicing.

Defining packet format

DAddress (1Byte)	SAddress (1Byte)	PLength (1Byte)	Packet (PLength Bytes)

DAddress [Destination Address]: It gives the destination address of the data to be send. It is one byte long .The maximum number of destination that can be addressed is 255.

SAddress [Source Address]: It gives the source addressed of the data to be send. It is one byte long.

PLength [Packet Length]: It gives the length of the data to send. The packet length field is one byte long.

Packet [Data]: This field gives the data to be send. The length of the data to be send is maximum up to 255 bytes.

© 2017, IJSRNSC All Rights Reserved

C. Algorithm for Sending Packet

- Initialize the pins
- Check for CS pin which indicates that if communication is already being done by another device
- If CS=0 bus is busy
  if CS=0 is found then return (0)
- If CS=1 bus is free
  indicating failure of comm...
- Capture the Bus my making CS =0
- Capture Dadd to the bus /rx
- Make SACK =0 indicating Dadd has been sent on bus
- Wait until Rx gives ACk or time out happens if timeout – then communication establishment failure
- Settle SACK
- Wait for Rx settlement
- Send SAdd on Bus
- Wait for H/S settlement
- Send Plength
- Wait for H/S settlement
- Send all the bytes in the packet & wait for H/S settlement
- Release the bus by making CS=1

D. Algorithm for Receiving Packet

- Initialize the pins
- Check for CS pin which indicates that if he Data is present on the Bus or not
- If CS=0, Data is present
- If CS=1;bus is free
- Wait till CS=0
- Wait till SACK=0
- Check whether Data on Bus= My Address
- Send RACK
- Wait for H/S settlement i.e. communication started
- Receive the sender's address from the Bus
- Wait for H/S settlement
- Receive the packet length from the Bus
- Wait for H/S settlement
- Receive the packets by executing a for loop (from 1 to packet length)
- Communication completed

E.The Time-Transition Diagram

A digital timing diagram is a representation of a set of signals in the time domain.

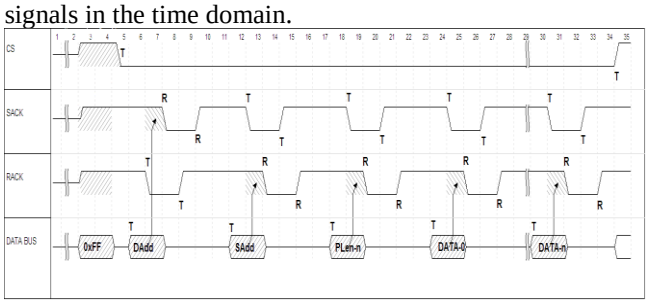

Fig. 3

V. CONCLUSION

In this paper, we have studied, Reconfigurable Multiprocessor Communication Protocol is more efficient protocol than I2C and SPI protocols. Here data transfer is parallel leading to a very high speed of data transfer. Synchronization is not necessarily required i.e. ICs can have different clock frequencies. Addition and removal of new controllers can be easily done without any modification of software and it has many more advantages than others.

Applications and Scope:

- In Network Communication for microcontrollers,
- In Onboard communication in embedded systems.

Limitations: As generally all systems have some limitation, here are some listed for the proposed system…

- This protocol can support maximum 255 nodes only,
- As it supports parallel data transfer and hence it requires more pins or lines for communication.

ACKNOWLEDGEMENT

I would like to express my gratitude to my supervisor Principal Dr.S.D.Sawarkar. I thank Dr. D.J.Pete for his guidance and discussions on this paper.

REFERENCES

[1]. Myke Predko, "*Programming and Customizing the 8051 Microcontroller"*, Tata McGraw-Hill, Inida, pp.157-167, 1999.

[2]. Shibu K.V., "*Introduction To Embedded Systems*" Published by Tata McGraw-Hill Education, Inida, pp.1-478, 2009,

[3]. Dogan Ibrahim, "*Microcontroller Projects in C for the 8051"*, Newnes, pp.29-161, 2008.

[4]. KJ. Ayala, "*The 8051 Microcontroller Architecture, Programming And Applications"*, WEST PUBLISHING COMPANY, USA, pp.131-197, 1993.

[5]. Michael J. Pont, *Embedded C*, Edition 2002, Addison Wesley, Page: 57-87,217.

[6]. N. Lekic, Z. Mijanovic, D. Gobovic, R. Dragovic-Ivanovic, "*The simple multiprocessor communication system*", 9th International Conference on Electronics Circuits and Systems, Dubrovnik, pp.1039-1042, 2002.

[7]. S. Iyengar, N. Apte, A. A. Roy, S. Sanyal, N. M. Singhi, WuGeng Feng, "*A multiprocessor communication architecture for high speed networks*", 1993 IEEE Region 10 Conference on Computer Communication Control and Power Engineering, Beijing (China), pp.262-265, 1993.

[8]. Karan Patel, Suraj Panigrahiya, Amit Pawar, Akash Survase, "*Opencv Based Virtual Touch Screen For Robotic Navigation*", International Journal of Students Research in Technology & Management, International Journal of Students Research in Technology & Management, Vol.2, Issue.6, pp.196-198, 2015.

© 2017, IJSRNSC All Rights Reserved