International Journal of Electrical, Electronic and Communication Sciences
ISSN: 2517-9438
Vol:2, No:11, 2008

An Efficient Hardware Implementation of Extended and Fast Physical Addressing in Microprocessor-Based Systems using Programmable Logic

Mountassar Maamoun, Abdelhamid Meraghni, Abdelhalim Benbelkacem, and Daoud Berkani

***Abstract*—**This paper describes an efficient hardware implementation of a new technique for interfacing the data exchange between the microprocessor-based systems and the external devices. This technique, based on the use of software/hardware system and a reduced physical address, enlarges the interfacing capacity of the microprocessor-based systems, uses the Direct Memory Access (DMA) to increases the frequency of the new bus, and improves the speed of data exchange. While using this architecture in microprocessor-based system or in computer, the input of the hardware part of our system will be connected to the bus system, and the output, which is a new bus, will be connected to an external device. The new bus is composed of a data bus, a control bus and an address bus. A Xilinx Integrated Software Environment (ISE) 7.1i has been used for the programmable logic implementation.

***Keywords*—**Interfacing, Software/hardware System, CPLD, programmable logic, DMA.

I. Introduction

THE inputs and the outputs are generally handled in the form of bytes, words, or double words for the most recent processors. The physical communication between an auxiliary circuit and an external card is ensured by enabling the address decoder as soon as the address of this latter card is available on the system bus. This mechanism explains why the installation of expansion cards causes sometimes conflict problems: it happens that two cards assert the same field of addressing or overlapping fields [1].

The Fast Physical Addressing is an interfacing system which is aimed to reduce the use of physical addresses in microprocessor-based systems. Furthermore, it will improve the data exchange speed compared to the Extended Physical Addressing technique [2]. The proposed system combines a software/hardware solution to obtain the above advantages. This solution consists in creating a new bus, made up of a data bus, an address bus and a control bus. The suggested architecture is composed of a hardware part and a software part. The first is made up of a new bus and an interface between the system bus and the new bus. The software part ensures the communication between the microprocessor-based system and our interface. In this technique we use the Direct Memory Access (DMA) to increase the data exchange speed between the microprocessor-based system and our system.

Manuscript received October 12, 2006. This work was supported in part by the LSIC Laboratory, (ENS) Kouba, Algiers, Algeria and the Signal & Communications Laboratory, (ENP) Algiers, Algeria.

M.Maamoun is with the Department of Electronic, Blida University, Blida, Algeria (e-mail: mountassar..maamoun@ens-kouba.dz).

A. Meraghni and A. Benbelkacem are with LSIC Laboratory, (ENS) Kouba, Algiers, Algeria (e-mail: meraghni@ens-kouba.dz).

D. Berkani is with the Signal & Communications Laboratory, (ENP) Algiers, Algeria.

II. Extended Physical Addressing

The Extended Physical Addressing is based on a mixed software/hardware architecture, which is intended to increase the addressing capacity of the computer and the microprocessor-based system [2]. In this system, we have used some addresses of the microprocessor-based system addressing area to address a larger external memory capacity. The material part of this system is made up of the new bus and the interface, which is between the system bus and the device that will be addressed by this technique. The input of this interface will be connected to the microprocessor-based system bus, and the output which is a new bus will be connected to an external device. The new bus contains a data bus and an address bus. Fig. 1 illustrates the basic idea of the hardware part.

The mechanism of this technique can be described in two steps. First, the software ensures the availability of the data intended to be present on the new bus as addresses. The decoder enables the D "LATCH" circuit to record these data and disables the two data bus buffers until they reach the second step. The address and data lines of the new bus will be at the high impedance state. Second, the software ensures the availability of the data.

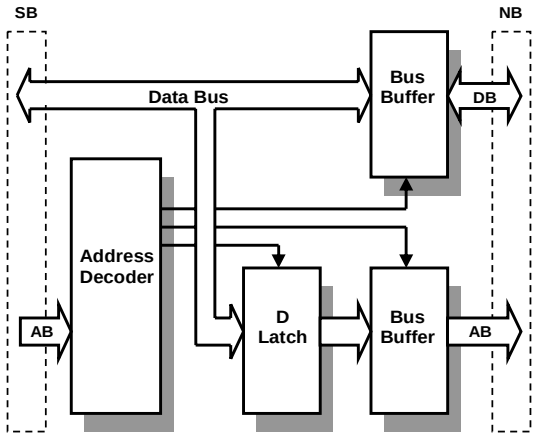

Fig. 1 Extended Physical Addressing bloc diagram

AB: Address Bus, DB: Data Bus, NB: New Bus, SB: System Bus

The address decoder enables the two bus buffers. The data values on the new bus are identical to the system data bus at this level. The addresses values are identical to the first step data of the system bus.

III. Extended Physical Addressing Implementation

In order to test the Extended Physical Addressing system, CPLD implementation is performed using Xilinx Integrated Software Environment (ISE) 7.1i. The Engineering Capture System (ECS) is exploited to create the schematic [3]. An XC9572-10PC84 device is used [4].

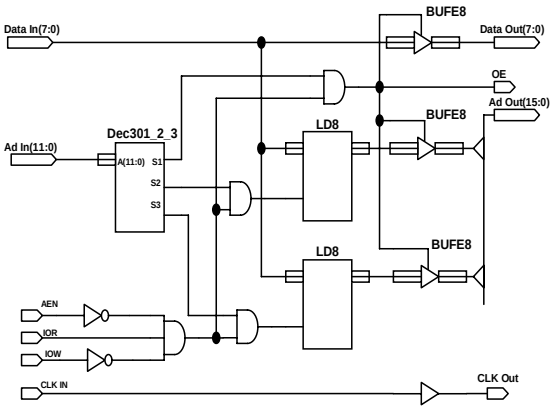

Fig. 2 ECS Extended Physical Addressing schematic

A model of schematic implementation is presented in Fig. 2 and the VHDL instantiation template is presented in Fig. 3 BUFE8 is an internal 3-State Buffer with Active High Enable and LD8 is a multiple Transparent Data Latch. The dec301_2_3 symbolize the address decoder. The entity and the architecture description of the address decoder are shown in Fig. 4.

```
Vhdl instantiation template
  COMPONENT epa01_sch
  PORT( Ad_In : IN STD_LOGIC_VECTOR (11 DOWNTO 0);
    AEN : IN  STD_LOGIC;
    Data_In : IN STD_LOGIC_VECTOR (7 DOWNTO 0);
    IOR : IN STD_LOGIC;
    IOW :  IN STD_LOGIC;
    OE : OUT STD_LOGIC;
    Data_Out : OUT STD_LOGIC_VECTOR (7 DOWNTO 0);
    Ad_Out : OUT   STD_LOGIC_VECTOR (15 DOWNTO 0);
    CLK_In : IN STD_LOGIC;
    CLK_Out : OUT STD_LOGIC);
  END COMPONENT;
  UUT: epa01_sch PORT MAP(
      Ad_In => ,
      AEN => ,
      Data_In => ,
      IOR => ,
      IOW => ,
      OE => ,
      Data_Out => ,
      Ad_Out => ,
      CLK_In => ,
      CLK_Out =>
  );
```

Fig. 3 VHDL instantiation template of the proposed Extended Physical Addressing implementation

```
entity dec301_2_3 is
  port (A : in std_logic_vector (15 downto 0);
        S1 : out   std_logic;
        S2 : out   std_logic;
        S3 : out   std_logic);
end dec301_2_3;

architecture DESCRIPTION of dec301_2_3 is
begin
   S1 <= '1' when A= x"0301" else '0';
   S2 <= '1' when A= x"0302"  else '0';
   S3 <= '1' when A= x"0303"  else '0';

end DESCRIPTION;
```

Fig. 4 Entity and architecture description of the proposed Extended Physical Addressing address decoder

IV. Fast Physical Addressing

This method uses the same technique of the data/address conversion, illustrated in the above section, for the address production of the new bus [5]. However, it uses a different process for the data transfer from the microprocessor-based system towards the input of the hardware part of the system. The input of the interface will be connected to the microprocessor-based system bus and the output will be connected to an external device.

Fig. 5 represents the Fast Physical Addressing bloc diagram. In this version, the hardware part explores the Direct Memory Access for the data exchange; consequently, the addresses on the new bus will depend on the data exchanged with the microprocessor-based system memory.

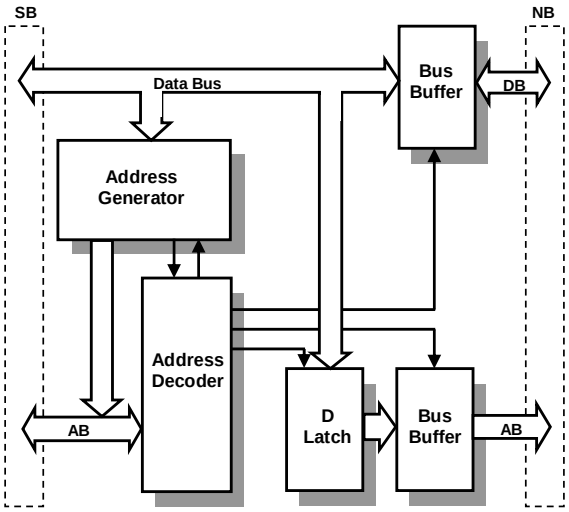

Fig. 5 Fast Physical Addressing bloc diagram

AB: Address Bus, DB: Data Bus, NB: New Bus, SB: System Bus

The software part of Fast Physical Addressing ensures two additional tasks. The first is to send the starting and ending positions of the accessible memory area. The second is to control the start and the end of the Direct Memory Access. The two positions of the memory area, which is addressed by the system, will be sent to the Address Generator AG on the data bus of the system bus on distinct addresses.

The DMA process of the system starts at these two data transfer. The address decoder enables the recording of these two data in the address generator circuit of the DMA. Once the process of the Direct Memory Access is started, the address generator controls the address bus of the system bus to carry out the reading or the writing on the selected memory area and enables the address decoder of the system to perform the decoding of the generated addresses. The address decoder ensures the distinction between the two types of data.

V. Fast Physical Addressing Implementation

The proposed programmable logic implementation of Fast Physical Addressing using Xilinx (ECS) is shown in Fig. 5. A CPLD implementation with Xilinx Integrated Software Environment (ISE) 7.1i is presented. An XC95108-15PC84 device is used [4]. The main schematic of the proposed implementation is shown in Fig. 6 and the VHDL instantiation template is presented in Fig. 7. The entity and the architecture description of the address decoder are shown in Fig. 8. The schematic and the entity of the proposed Address Generator are illustrated respectively in Fig. 9 and Fig. 10.

BUFE16 is an internal 3-State Buffer with Active High Enable and LD16 is a multiple Transparent Data Latch. The fpa_dec00 represent the Fast Physical Addressing address decoder and ag01 represent Address Generator.

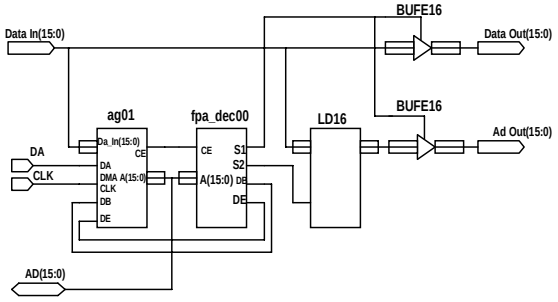

Fig. 6 ECS Fast Physical Addressing schematic

```
Vhdl instantiation template

COMPONENT fpa00
PORT(AD : INOUT STD_LOGIC_VECTOR (15 DOWNTO 0);
     Data_In     : IN STD_LOGIC_VECTOR (15 DOWNTO 0);
     DA          : IN   STD_LOGIC;
     CLK         : IN STD_LOGIC;
     Data_Out    : OUT    STD_LOGIC_VECTOR (15 DOWNTO 0);
     AD_Out      : OUT STD_LOGIC_VECTOR (15 DOWNTO 0));
END COMPONENT;

UUT: fpa00 PORT MAP(
     AD =>,
     Data_In =>,
     DA =>,
     CLK =>,
     Data_Out =>,
     AD_Out =>
);
```

Fig. 7 VHDL instantiation template of the proposed Fast Physical Addressing implementation

```
entity fpa_dec00 is
   port(
        CE : in std_logic;
        A : in std_logic_vector(15 downto 0);
        DB : out std_logic;
        DE : out std_logic;
        s1 : out std_logic;
        s2 : out std_logic);
end fpa_dec00;
architecture DESCRIPTION of fpa_dec00 is
begin
    process(CE,A)
    begin
        if ((CE='0' and A=x"0301") or (CE='1' and A(0)='1')) then
            s1 <= '1';
            else
            s1 <= '0';
    end if;
        if((CE='0' and A=x"0302") or (CE='1' and A(1)='1'))then
            s2 <= '1';
            else
            s2 <= '0';
    end if;
    if(CE='0' and A=x"0303") then
            DB <= '1';
            else
            DB <= '0';
    end if;
    if(CE='0' and A=x"0304") then
            DE <= '1';
            else
            DE <= '0';
    end if;
```

```
    end process;
end DESCRIPTION;
```

Fig. 8 Entity and architecture description of the proposed Fast Physical Addressing address decoder

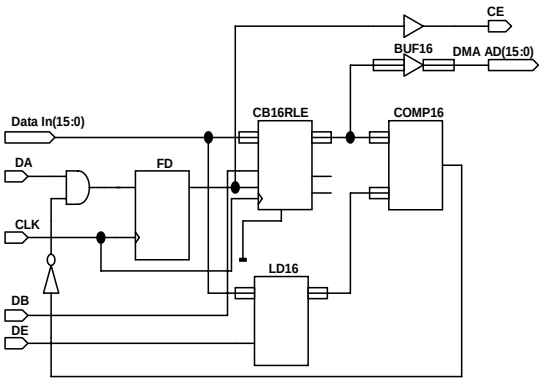

Fig. 9 Schematic of the proposed Address Generator

```
entity ag01 is
  port (CLK    : in std_logic;
        DA     : in std_logic;
        Data_In : in std_logic_vector (15 downto 0);
        DB     : in  std_logic;
        DE     : in  std_logic;
        CE     : out  std_logic;
        DMA_AD  : out std_logic_vector (15 downto 0));
end ag01;

architecture BEHAVIORAL of ag01 is
begin
end BEHAVIORAL;
```

Fig. 10 Entity of the proposed Address Generator

VI. Results

The application of this version on a system bus that works with a data bus of N bits and a maximum frequency F, gives a new bus of N bits of data, and has an addressing physical capacity of 2^N. This new bus works at a maximum frequency equal to Fn. Table 1 represents the main characteristics of Fast Physical Addressing. The maximum frequency of the new bus is described by the following formula.

$$Fn = \frac{1}{2 \times Ta}$$

where Ta, is the memory access time used during the DMA process, or the memory access time used in our system.

TABLE I
Characteristics of Fast Physical Addressing

FI	NDS	NDN	NAN	FO
F	8 bits	8 bits	256	Fn
F	16 bits	16 bits	64 K	Fn
F	32 bits	32 bits	4 G	Fn

FI: System bus work frequency
FO: New bus work frequency
NAN: A number of physical addresses available on the new bus
NDN: new bus data size.
NDS: system bus data size

For example, if the Fast Physical Addressing is applied to a system bus, which runs at a 66 MHz frequency, a data bus of 32 bits and a 6ns access time of the used memory, the new bus will work with a physical addressing capacity of 4G and a maximum of 83 MHz work frequency

VII. Conclusion

In this paper, we have presented a hardware implementation of the Extended and the Fast Physical Addressing with programmable logic using Xilinx CPLD devices. This solution is based on the use of mixed software/hardware system. The Extended Physical Addressing system, which represents the starting point of our system, is exposed at the beginning. Thereafter, we presented the solution of the Fast prototype. This new system presents a solution for improving the physical addressing capacity of the microprocessor-based systems, which is independent of the addresses bus of the system bus. The main technique of this interfacing system has been applied for the implementations of prototype cards which work on IBM™ computers and compatible [6].

Currently, we are investigating the option of adapting and implementing this system in PCI bus using CPLD and FPGA devices.

References

[1] M. Tischer. “La Bible PC”. Editions Micro Application. France. 1997
[2] M. Maamoun, G.Zerari. “Adressage Matériel dans les Systèmes à Microprocesseur avec un Adressage Physique Etendu”. 2001 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE01). Toronto, Ontario, Canada. 2001.
[3] Xilinx ISE 7.1i Software.
[4] Xilinx Product Specification, 2003, www.xilinx.com.
[5] M. Maamoun, A.Benbelkacem, D.Berkani, A.Guessoum “Interfacing in Microprocessor-based Systems with a Fast Physical Addressing”, The 3rd IEEE International Workshop on System-on-Chip for Real-Time Applications. Calgary, Alberta, Canada. 2003
[6] M. Maamoun. “Conception et Réalisation d’un Système Mixte Logiciel/Matériel pour l’Affichage des Images PC sur un Moniteur TV”. Thèse de Magister. Université de Blida. Algérie. 2000.