SERIAL PORT INTERFACE FOR MICROCONTROLLER EMBEDDED INTO INTEGRATED POWER METER

Borisav Jovanović, Predrag Petković, Milunka Damnjanović

Laboratory for Electronic Design Automation, Faculty of Electronic Engineering, University of Niš, Aleksandra Medvedeva 14., 18000 Nis, Serbia, phone: +381 18529321, e-mail: borko@venus.elfak.ni.ac.yu

This paper presents some of the peripheral units embedded into microcontroller within Integrated Power Meter which represents mixed signal ASIC dedicated for power-metering. The microcontroller block is compatible with 8052 microprocessors and processes most instructions in one cycle. Peripherals include two programmable UART modules, three 8-bit wide digital input-output ports and I^2C-like serial port interface for communication with an external EEPROM memory. Besides, the 8052 microcontroller incorporates real-time clock module and LCD driver circuits capable to support up to 168 pixels LCD. The paper describes design of the peripherals supported by the microcontroller together with the methods for chip programming and program memory initialization.

Keywords: Microcontroller, Integrated, Power, Meter

1. INTRODUCTION

The lack of clean energy becomes an important subject that provokes scientists from different fields to cope with the problem from different aspects. One of important links in the chain of energy managing is accurate and efficient measurement of produced and spent power. Modern power meters relays on single chip devices referred to as *integrated power meter* (IPM). There are numerous IPM solutions that consider one-phase [1, 2] and three-phase power meter [3]. This paper considers an original design of a three-phase IPM that integrates all primary functional blocks required to implement solid-state energy meter. It is a mixed signal circuit consisting of analog and digital signal processing blocks. The signal processing chain is given in Fig.1.

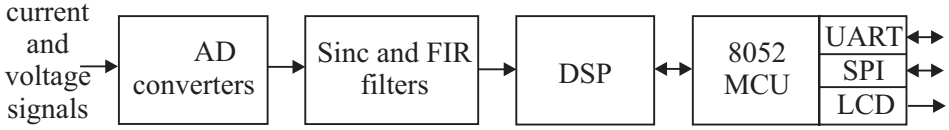

Fig.1 Architecture of the Integrated Power-Meter

The analog part of IPM contains Sigma-Delta AD converters for current and voltage channels [4], Band-Gap voltage reference and PLL circuits. The digital part is composed of digital filters, digital signal processing block (DSP) and microcontroller unit together with integrated peripherals that are the subject of this paper.

The digital filters [5] decimate oversampled output signals from the on-chip AD converters for both voltage and current signal channels in three phases.

The DSP performs the precision computations necessary to measure: active, reactive and apparent energy in four quadrants for all three-phases, instantaneous

frequency for each phase, RMS currents and voltages, active, reactive and apparent power and power factor [2].

The microcontroller unit (8052 MCU shown in Fig.1) is compatible with 8052 microprocessors. It includes several communication peripherals: UART, *serial port interface* (SPI) and LCD driver circuit.

This paper describes functional level design of peripheral modules needed for communication between MCU and its environment. Although the functional level is mostly technology independent, it is not the case for this project because the available technology (AMIS CMOS 0.35μm, [7]) does not support on-chip non-volatile memory blocks. Therefore at functional level some important decision about data storing has to be made in order to suppress losing data. The crucial one is using external EEPROM devices together with the Integrated Power Meter.

This paper is organized in five sections and References. The following section gives an overview of MCU. The third section considers serial port interface unit dedicated for communication between IPM and external EEPROM. The subsequent section describes MCU programming and initialization. The fifth section gives concluding remarks.

2. THE MICROCONTROLLER UNIT OVERVIEW

The microcontroller unit implements fetch and execution phases in parallel. One-byte instructions are performed in a single cycle. The using of 4.194 MHz clock results in a processing throughput of more than 4 MIPS. The microcontroller memory organization is similar to those of industry standard 8052 microcontrollers. There are three memory areas associated with the microcontroller illustrated in Figure 2.

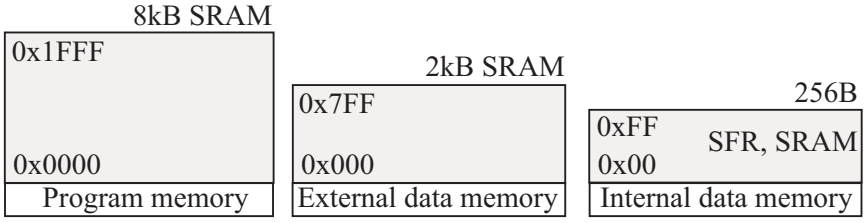

Fig.2 Microcontroller unit memory map

Namely they are

- program memory, implemented as internal 8kB SRAM block;
- external data memory, 2kB SRAM memory block which also resides on chip;
- internal data memory shared between Special Function Register (SFR) and 256B RAM .

Maximal flexibility of MCU is supported with two programmable UART modules, three 8-bit wide digital in-out ports and serial port interface for communication with external EEPROM memories that is the topic of this paper.

The microcontroller unit controls operating mode of DSP driving appropriate states to DSP inputs. Beside, special function registers are used to access to all internal registers of DSP that contain important information about many DSP calibration parameters [6] and various power signal measurement results that DSP provides [2] .

After resetting MCU, program memory is automatically loaded from external EEPROM memory through SPI pins. The method of program memory initialization will be explained in the fourth section. Beside, the IPM has ability to receive MCU program through UART input pin and store it into external EEPROM memory. A special programming mode controls the EEPROM programming.

3. SERIAL PORT INTERFACE WITH EXTERNAL EEPROM DEVICE

Communication between MCU and the external EEPROM requires data transfer during initialization, programming and normal operation mode. During initialization it is necessary to read program from EEPROM into on-chip SRAM. In programming mode data received from UART module are written to the EEPROM. Besides, during normal operation mode, measured and calculated power line parameters should be stored in EEPROM occasionally. Access to EEPROM requires serial communication. Actually, the communication between MCU and EEPROM is designed as an I2C-like approach referred to as serial port interface. Hardware module dedicated for this communication purpose is named MASTER SPI.

External communication, to EEPROM requires only two pins: *SDA* for data and *SCL* for SPI clock. The serial clock frequency is 100 kHz. Internal communication with MCU employs two 8-bit Special Function Registers: EEDATA and EECTRL that are embedded into MASTER SPI block.

Contents of EEDATA and ECTRL registers define communication with the external EEPROM. The four most significant bits of EECTRL register define the status of SPI communication (the line is busy, acknowledge bit was transmitted, acknowledge bit was received, illegal command was written into 4 LSB bits of EECTRL). Writing data into four least significant bits of EECTRL register commences the MASTER SPI actions. The corresponding command values and MASTER SPI actions are given in Table 1.

Bit position	Name	Description
7	Error	Illegal command received
6	Busy	Serial Data Bus is busy
5	Rx_ack	Acknowledge received from EEPROM
4	Tx_ack	Acknowledge transmitted to EEPROM
3:0	Cmd(3:0)	2 Receive a byte from EEPROM and send ACK 3 Transmit a byte to EEPROM 5 Issue a STOP sequence 6 Receive a byte from EEPROM and do not send ACK 9 Issue a START sequence others Illegal command

Table 1. EECTRL register content

MASTER SPI block operates as follows. If a byte of data is to be transmitted to

the EEPROM, that byte first has to be stored into EEDATA register. Writing value Cmd=3 (see Table 1) into EECTRL register starts *Transmit a byte to EEPROM* process. By observing the sixth bit in EECTRL a control unit of MCU receives information whether the line is busy or not. When *busy*=1, transmission is in progress; busy=0 denotes that the transmission is over and the line is free. The fifth bit of EECTRL is set if EEPROM acknowledged the transmission.

A data read operation starts by writing the *Receive a byte from EEPROM and send ACK* command (Cmd=2, see Table 1) to EECTRL register. The operation is finished when *busy* signal change value from 1 to 0 after the ninth SCL rising edge. Thereafter received data remains in EEDATA register.

Start and stop sequences on SPI bus lines can be issued by writing the appropriate commands to the EECTRL register. When detected *Issue a START sequence*, Cmd=9 in EECTRL register, SDA line is pulled from high to low while SCL remains high, that is default when SPI is not in use. However, with Cmd=5 command *Issue a STOP sequence* occurs, putting SCL in high logic state and turning SDA line up after.

4. MCU PROGRAMMING AND INITIALIZATION

Proper IPM operation requires options for MCU programming and initialization. During the programming mode, the program is loaded through UART module and transferred into external 24LC64 EEPROM [8]. The initialization mode begins after MCU reset. Then the EEPROM content is moved into on-chip SRAM memory block. The hardware used to store the program in EEPROM and to load the on-chip SRAM with EEPROM content is given in Fig.3.

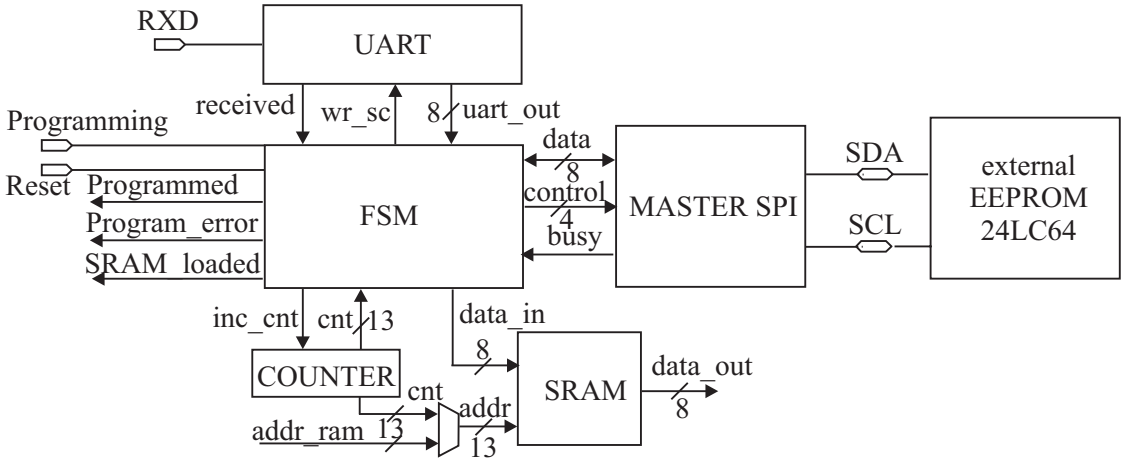

Fig. 3. The part of chip dedicated for EEPRPOM programming and SRAM initialization

It consists of the following digital blocks: MASTER SPI, UART, Finite State Machine (FSM) that controls the SPI communication process, SRAM block and 13-bit COUNTER that defines the address for SRAM.

Chip programming mode is enabled by setting input pin *Programming* at high level. After reset, in programming mode, all data received from UART block (Fig.3) are serially transmitted in form of 32-byte data packages through *SDA* pin to the EEPROM device. Figure 4 illustrates the structure of one data package sent to the EEPROM during programming mode.

The package begins with *start* bit, followed by *device address* byte, *high memory address* byte, *low memory address* byte, 32 *data bytes* and ends by *stop* bit. Bytes are separated by *acknowledge* bit.

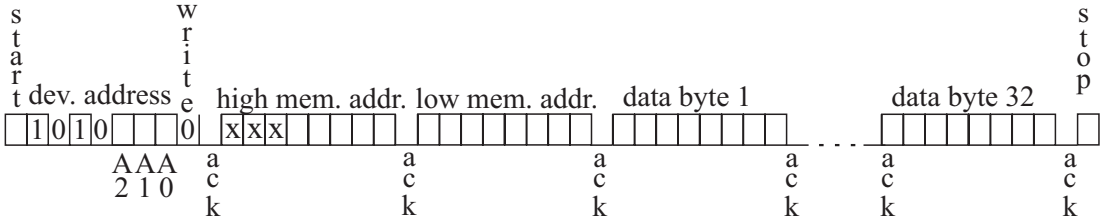

Fig. 4 The structure of one data package during programming mode

The FSM controls the whole programming process by sending the commands to the MASTER SPI block.

To send a 32-byte data packet, FSM must write Cmd=9 into EECTRL issuing START sequence. Thereafter, FSM sends to EEDATA the device address of the selected EEPROM. The device address consists of eight bits 1010(A2)(A1)(A0)0 as shown in Fig.4. The four most significant bits of the address byte (1010) code the EEPROM device. Bits A2, A1 and A0 define the selected specific EEPROM device (one of up to eight connected to SPI bus). LSB address bit determines whether to perform write (0) or read (1) operation.

Data transfer starts when command *Transmit a byte to EEPROM* (Cmd=3) is written into EECTRL.

After transmitted device address to the EEPROM, FSM receives from EEPROM acknowledge bit on the ninth rising edge of SCL. The FSM sends high memory address byte, low memory address byte and 32 data bytes. Each *send* operation starts by writing *Transmit a byte to EEPROM* command setting Cmd=3 into EECTRL. At the end of every data package, FSM sends *Issue a STOP sequence* command when 32 bytes are permanently stored in the EEPROM. The stop sequence is commenced by writing the Cmd=5 into EECTRL.

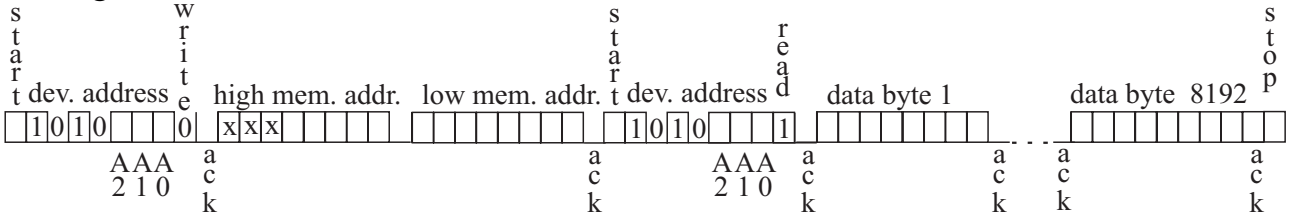

Fig.5 The structure of data package during SRAM initialization mode

After sending 256 32-byte data packages, FSM ends programming operation and change the state of output pin *Programmed* (Fig.3) to 1. If UART receive wrong stop bit during programming operation, which means that an error occurred during transmission, programming operation is aborted and the output pin *Program_error* changes to 1.

Chip initialization mode is enabled by holding the input pin *Programming* in low state. Figure 5 shows the structure of data package needed for SRAM initialization mode.

Procedure begins after MCU reset. The FSM starts the communication with a dummy write operation (one with no data bytes) to load the EEPROM address

counter. FSM first sets Cmd=9 in EECTRL to *Issue a START sequence.* Then sends the device address byte 1010(A2)(A1)(A0)0, followed by the high and low memory address bytes. After receiving the acknowledge, FSM starts the read operation by issuing the new start sequence, followed by new read device address byte 1010(A2)(A1)(A0)1. The LSB bit in the second device address byte signifies read operation. The EEPROM responds with acknowledge. Afterwards it serially shifts out the data byte from the location that corresponds to the current memory address counter value. The data received from EEPROM is stored into SRAM memory. FSM sends acknowledge indicating that it requires additional data bytes and increments SRAM address counter (COUNTER shown in Fig.3). After receiving acknowledge from FSM, EEPROM increments its own memory address counter by one and shifts out the next data byte. The sequential read operation continues as long as FSM keeps acknowledging. When SRAM address counter points to the last SRAM memory location, FSM reads the last data byte from EEPROM and terminates the read operation by not acknowledging and writing Cmd=5 into EECTRL.

5. CONCLUSION

The digital block MASTER SPI dedicated for communication with external EEPROM device is considered in this paper. The described block utilizes I2C-like interface and it is a part of a 8052 microcontroller which is embedded into Integrated Power Meter System-On-Chip. MASTER SPI can be programmed by 8052 microcontroller through two 8-bit Special Function Registers EEDATA and EECTRL.

The application of proposed hardware dedicated for chip initialization and programming is considered, as well. The piece of hardware that controls MASTER SPI in programming and chip initialization mode is implemented as a finite state machine.

ACKNOWLEDGE

Results described in this paper are obtained within the project TR6108 founded by Serbian Ministry of Science.

6. REFERENCES

[1] ADE7169F16, Single-Phase Energy Measurement IC, Analog Devices, 2006

[2] Jovanovic B., Damnjanović M., Petković P., *Digital Signal Processing for an Integrated Power Meter*, Proceedings of 49. Internationales Wissenschaftliches Kolloquium, Technische Universirtat Ilmenau, Ilmenau, Germany, vol. 2, pp. 190-195, September 2004

[3] 71M6513, 3-Phase Energy Meter IC, TDK Semiconductors, 2006

[4] Milovanović D., Savić M., Nikolić M., *Second-Order Sigma-Delta Modulator In Standard CMOS Technology*, Proc. of XLVIII Conference ETRAN, pp. 17-20, 2004, Čačak, Serbia

[5] Marinković M., Anđelković B., Petković P., *Compact Architecture of Digital Decimation Filters in Solid-State Energy Meter*, Proc. of VI Simposum INDEL, pp. 84-89, 2006, Banjaluka, Bosnia and Hercegovina

[6] Damnjanović M., Predrag Petković P., Jovanović B., *Integrated Power Meter IC Calibration,* Proc. of Small Systems Simulation Symposium, pp. 59-62, March 2005, Niš

[7] AMI Semiconductor CMOS 0.35μm Technology Documentation

[8] 24LC64 EEPROM Memory IC Documentation, Microchip, 2005