Simple parallel stream to serial stream converter for Active Pixel Sensor readout.

V. Kushpil[1], M.Šumbera[1], M.Szelezniak[2].

[1] Nuclear Physics Institute ASCR, 25068 Řež/Prague, Czech Republic
[2]Berkeley National Laboratory, 1 Cyclotron Rd. Berkeley, CA 94720, USA

kushpil@ujf.cas.cz

Abstract

This paper describes a new electronics module for converting a parallel data flow to a serial stream in the USB 2.0 High Speed protocol. The system provides a connection between a PC USB port and a parallel interface of the DAQ board, which is used for investigation of performance of Active Pixel Sensors (APS) prototypes. The DAQ readout software supports Win XX OS and Linux OS. GUI examples have been prepared in the Lab Windows and Lab View environments. The module that was designed using virtual peripheral concept can be easily adapted for many similar tasks.

I. INTRODUCTION

High Granularity Semiconductor Detectors (HGSD) (pixel, micro strip and drift) are a powerful tool in high-energy physics. Readout electronics for HGSD is manufactured as ASIC chips (contained preamplifiers, shapers, analog and digital memory and ADC) that can be controlled by FPGA based circuits [1]. The Custom-build Modules Readout (CMR) is used to handle the transfer of data between the HGSD and the main computer for data storage. For example, for investigation of Active Pixel Sensor (APS), the LBL APS group uses a simple parallel data transfer protocol with readout rates of about 60MB/s [2]. Data from DAQ are sent to PC synchronously with Process Clock (PCLK) signal and the data flow is controlled by REQ (request) and ACK (acknowledge) signals.

The main disadvantage of CMR is that the readout system requires a digital DAQ PCI card that needs to be installed inside a PC and this limits the portability of the system. Also the multi-conductor SCSI-like cable limits to some extent the portability of the system.

In this paper we described a simple, 16 bit parallel to USB 2.0 stream converter which allows readout with data rates of about 48 Mbytes per second and can be easily adapted to many different readout architectures and different OS (WinXP, Linux). Flexibility of the converter is achieved by using the virtual peripheral concept [3] for design and fastest 8-bits micro controller SX28 from UBICOM [4]. By using this module, the APS DAQ can be connected to a portable computer allowing the use of different Operating System (OS) with the same hardware (HW) and software (SW).

II. HARDWARE

As shown in Fig.1, the Parallel to Serial Flow Converter (PSFC) consists of three main parts: MCU control, FIFO memory and Quick USB (Q-USB) module. The converter module is designed for high performance and maximum flexibility. It contains the single chip of FIFO memory, single chip of the micro controller SX28, three chips of digital buffers and one Q-USB module. The converter consists of 16 bits parallel input Din[15..0], input lines REQ and PCLK and output line nFULL (FIFO is full). The FIFO memory is a CMOS chip CY7C4506 (16KB x 18 bits) operating with 100 MHz clock (this chip reads and writes data on the front edge of the clock signal). The MCU SX28 control unit provides the data flow synchronization. To obtain acceptable processing time of conversion the simple control algorithm is used. The stages of conversion are described below.

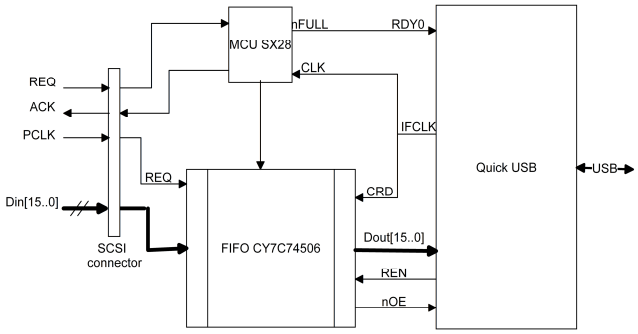

Figure 1: The block diagram of the Parallel to Serial Flow Converter (PSFC).

Stage 1: Read 16 bits word data from DAQ and store first 8KB data in FIFO memory. The 16 bit data from APS are sent synchronously with PCLK signal and the data flow is controlled by REQ and ACK signals. The signal REQ set to '1' informs converter that DAQ is ready to start the data transfer. In response to ACK='1', REQ is set to '0' and a data package from DAQ is sent synchronously with the PCLK clock signal.

Stage 2: After writing data to FIFO during 8KB/12MHz~0.7ms is starting the process conversion. After converter receives the data package (during 16KB/48MHz=~0.33ms, or 330/125~3 USB2 (High-Speed) frames), ACK is set to '0'. When signal REQ is set to '1' and when nFULL is equal '1' the MCU enables writing data to FIFO memory by setting the signal WEN (Write Enable).
This enables the Q-USB module to read data from FIFO and to send it to PC via USB. The Q-USB module operation is described in tutorial and will not be explained here. We will remark only that the Q-USB can be configured for data readout in different modes (master device, slave device, "data

tube" mode, full handshake FIFO mode and more…). The signal nFULL=0 switches the Q-USB module into a waiting mode. The FIFO memory is used as a temporary storage buffer if frequency PCLK is high then frequency IFCLK. The signal for synchronization of the beginning of data transfer is absent in the original APS DAQ. The synchronization is achieved by resetting DAQ when PSFC is in the waiting mode

III. Software and Results

Two types of software were developed for the converter. The first one is the converter software that was written in SX assembler code. Codes are very simple and can be easily modified. Modification of the code can be used for adjusting the number of data blocks used for transfer of a multiple frame from DAQ or for additional data processing during of the converter operation. For example, we can use the interrupt service to monitoring nFULL signal during high-speed data transfer from DAQ or software monitoring for that.

The second type of software is the readout software that was prepared for WinXP and Linux OS. The Quick USB has two important advantages. The first is that it can be used with different OS (Win XX, Linux and MAC). The second is that Q-USB supports include libraries of standard functions for different kind of compilers (MS Visual C, Borland CPP, Lab Windows, Lab View and GNU C). Short examples for all OS and compilers described above were prepared. The converter was tested with APS DAQ motherboard version V3.02 developed at LBL.

The converter module was tested by reading full frames of APS prototype MIMOSA5 [5], which consists of four sub-arrays of 512 x 512 pixels each. The DAQ is operating as master device and the converter is a slave device. The data were sent frame by frame. The maximal input data rate A can be define as A=M/T +B, where M- size of FIFO, B-data conversion rite, T- frame sending time. For M=0.16Mbyte, B=48MHz, T=75msec we can obtain A=50Mbyte/sec. From real test we can conclude that converter practically doesn't change output data rating. Specific delay is defined by OS. For WinXP the readout process must be run with high priority (level 13) to realize fast data flow conversion. The readout with converter was reliable and system operated continuously nearly 36 hours.

IV. Conclusions

A simple parallel to serial data stream converter for connection between APS DAQ parallel interface and USB port of PC was developed and tested. The readout DAQ software developed for this system supports Win XX OS and Linux OS. The module that was designed using virtual peripheral concept can be easily adapted for many similar tasks. The hardware description, Gerber files, and firmware for the converter module can be downloaded from: http://ojs.ujf.cas.cz/~kushpil/APS

V. Reference

1] E.J. Siskind, Data acquisition system issues for large experiments , Nucl. Instr. and Meth. A 579 (2007) pp.839–843

[2] LBL readout LBL readout schematic http://www.lbnl.leog.org/pdf/pixel_daq_motherboard.pdf

[3] SX Virtual Peripheral Methodology & Modules Rev. 1.0 © 2000 Scenix Semiconductor, Inc.

[4] SX28 User Guide 1.0 © 2000 Scenix Semiconductor, Inc.

[5] Schematic MIMOSA5 http://www.lbnl.leog.org/pdf/mimosa5_schem.pdf

This work was supported by the Ministry of Education of the Czech Republic (grants LA08015 and LC70480).