PIC32-Based Scheme for Signaling and Transferring Images From an FFT-CCD Sensor Through USB

A. Serrano[1], J.C. Vélez[2], M. Calle[3]

[1, 2, 3] Grupo de Investigación en Telecomunicaciones y Señales
Universidad del Norte
Barranquilla, Atlántico, Colombia
[1] *antonio.r.smendoza@gmail.com

ABSTRACT
The paper presents important design characteristics of the control scheme for an FFT-CCD sensor. The hardware module employs a microchip PIC32 core with a Hamamatsu CCD sensor, for image retrieval, conversion and transfer to a computer. The paper focuses on describing the core programming for sensor control and USB communication. The description provides useful information for researchers and hardware interface developers. The system was tested for transfer delay in USB connections with different number of endpoints and three endpoint sizes. Results show that USB connections with three endpoints and 64-byte endpoint size have the smallest transfer delay, 1 second. Connections with two endpoints increase delay in 100 milliseconds; however, this setup is simpler to implement.

Keywords: full frame transfer, CCD, USB, acquisition, control.

RESUMEN
El siguiente artículo presenta las características más importantes del diseño de un sistema de control para un sensor FFT-CCD. El módulo *hardware* utiliza un controlador PIC32 de microchip y un sensor CCD de Hamamatsu, para capturar, convertir y transferir una imagen a un computador. El artículo se enfoca en la descripción de la programación del PIC para el control del sensor y de la conexión USB. Esta descripción provee información útil para investigadores y desarrolladores de interfaces en *hardware*. El sistema se sometió a pruebas de tiempo de transferencia en conexiones USB con diferentes números y tamaños de *endpoints*. Los resultados muestran que las conexiones USB con tres *endpoints* con 64 bytes de tamaño permiten obtener el menor tiempo de transferencia, 1 segundo. Cuando se utilizan dos *endpoints*, el tiempo de transferencia aumenta en 100 milisegundos, pero la implementación es más sencilla.

1. Introduction

The main element in any support tool for digital imaging requires an optical sensor for converting light in electrical signals. Optical sensors are built with either charged coupled devices (CCDs) or complementary metal-oxide semiconductors (CMOS). Many devices based on these sensors present flexible connectivity to computers, allowing for fast acquisition, presentation, storage or distribution of sampled images. Most research in this field relates to materials, manufacturing techniques, and technologies associated with the optical sensor itself. However, another part of the digital imaging system must be considered: the electronic circuitry for control and for image acquiring, converting, transferring and processing. Most developing countries lack the necessary tools to develop state-of-the-art optical sensors. Nevertheless, engineers in these countries may build acquisition systems, which play a key role in optical sensing technology, particularly in medical imaging applications. The work was particularly oriented to design a control and transfer system for a full frame transfer (FFT) CCD x-ray intraoral sensor. Nonetheless, the designed scheme can be used with any other type of FFT-CCD sensor. For more information on the FFT variants of a CCD sensor, the reader is referred to [1, 2].

Several systems for sensor control and image transfer have been reported. The study in [5] presents a CMOS-based intraoral sensor using USB as transfer protocol. Bin and Liqiang [6] also

present a CMOS-based sensor with USB connectivity for endoscopy applications and video transfer. Other USB-based image acquisition systems, developed for general purposes, are presented in [7 - 9]. Studies in [10, 11] presented general-purpose systems based on the Ethernet protocol.

Unlike previous solutions, this paper describes a unified core for signaling and transfer. The solution minimizes space and also reduces noise vulnerability. Additional modules required are voltage adjustment and conditioning circuits, not covered in this document. The paper also presents key aspects of CCD sensor signaling as a useful resource for future developments.

2. Hardware description

The sensor used in this work is Hamamatsu S8980 [12], specifically designed for intraoral x-ray applications. The primary element of the sensor is an FFT-CCD with 20x20 μm^2 pixel size and a 1.5 megapixel resolution. The sensor does not contain driver circuits or embedded automatic controllers thus control signals must be generated from outside according to the desired output frequency. The sensor contains one horizontal shift register and 1001 vertical registers, which build up the sensitive area. FFT-CCD sensors lack of storage area, and all pixels belong to the active area. According to [2], and using the same notation, the timing signals required by any FFT CCD sensor are horizontal register phase signals (P1H, P2H), vertical phase signals (P1V, P2V), transfer gate (TG), summing gate (SG) and reset gate (RG).

Selection of the main core corresponds to sensor requirements and the need for a system as integrated and inexpensive as possible. The application required a core including the following characteristics: capability to generate CCD control signals, an ADC module with a sample rate of 1 megasamples per second (Msps), and USB connection. Field programmable gate arrays (FPGAs) and complex programmable logic devices (CPLDs) were discarded because they do not integrate any of the required modules, forcing the addition of external systems. Digital signal processors (DSPs) were discarded because the scheme presented in this paper does not perform complex operations on acquired images. We decided to use a core of the microchip PIC32 family, more specifically, PIC32MX440F128H [13]. The device features 80MHz top clock frequency, embedded USB 2.0 module designed for full speed transfers (12Mbps), integrated 1 Msps analog-to-digital converter (ADC) module and direct memory access (DMA) modules for automating memory transfer tasks. PIC32 also provides five timers, with up to 32-bit length period, running synchronously with the core or asynchronously with an external input [13]. The core was programmed to generate control signals required by the CCD sensor.

Universal serial bus (USB) technology was chosen for connecting the sensor to a computer because it is flexible and widely spread in both commercial and research applications. Technology is affordable and the USB protocol is simple and provides secure information transfer to and from the computer [3, 4].

A USB module functioning at full speed achieves 12 Mbps data rate, slower than USB high speed data transfer rate (480 Mbps). However, considering that image size is close to 1.5 megabytes, nominally it would take about one second to transfer the image through a 2.0 USB port; hence, no higher data transfer rate is required.

3. CCD control signal generation and USB management

The core executes the following main tasks: generation of CCD control signals, control and monitoring of USB connection and control of ADC data. Among all tasks, the generation of CCD control signals has the most critical timing requirements, thus this task is handled by top priority interrupts. The core manages all USB tasks through low priority interrupt service routines and transfer control is shared between the normal flow of the program and the interrupt routine related to CCD control signals.

Figure 1 shows the time diagram for the specific sensor implemented in this work. The timing characteristics agree with the manufacturer's requirements: pulse width for P1V, P2V and TG is 60µs. P1H, P2H and SG have equal pulse

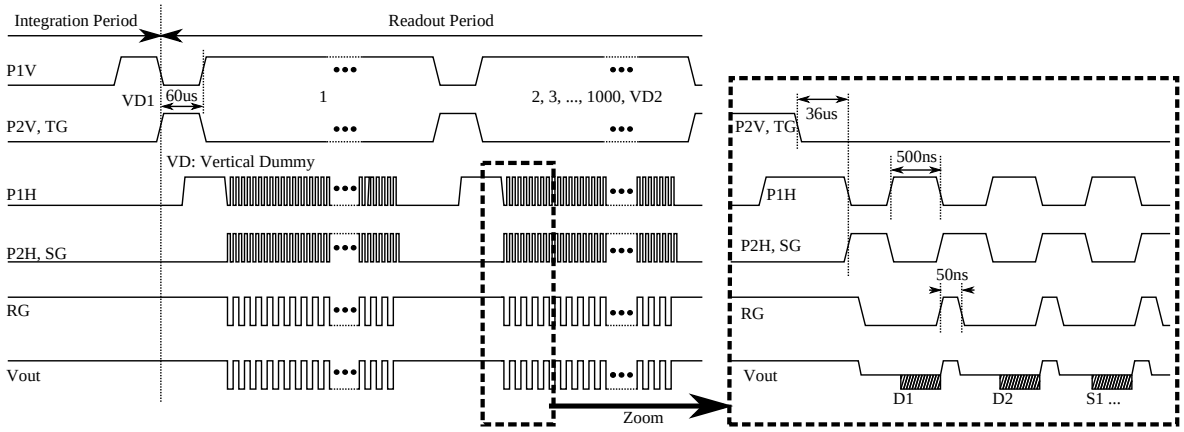

Figure 1. Timing scheme to control FFT-CCD based image sensor, according to manufacturer specifications.

widthaccording to ADC sampling rate; the manufacturer recommends 500 ns, but final implementation used 600ns. RG employed 50 ns pulse width.

The core generates vertical register phase signals by toggling the correspondent port register bits in memory. Also, horizontal register phase signals had more strict timing requirements thus the core generates them by using timers and output compare (OC) modules. Reset signal is also generated by means of OC modules. One of the embedded timers, running synchronously with the core, managed all operations related to signal generation. This timer controls a maximum priority interrupt with hardware context saving, thus reducing interrupt handler latency time. Figure 2 shows the state diagram used for signal generation.

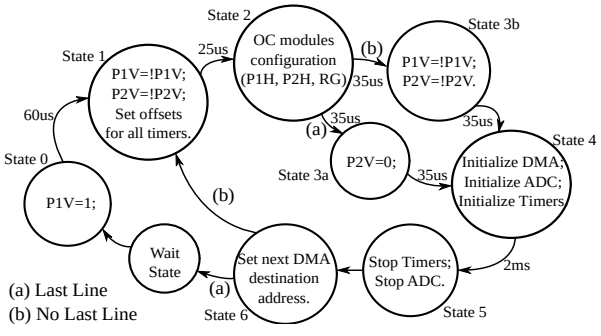

Figure 2. State diagram for CCD signal generation. Labels next to each line are delays between states.

Control signal generation is as follows: PIC32 Timer 1 controls state transition and its period is reassigned in each state during the interrupt service routine according to the values shown in Figure 2. OC modules control generation of P1H, P2H and RG (the fastest signals). The core assigns P1H and P2H OC modules to the same timer but operation modes are inverted to accomplish 180º shifts between them. The process starts when the computer commands the PIC32 to retrieve information from the CCD. During this first step, the core sets P1V to high, configures and initiates Timer 1. Once the timer period is completed (after 60µs approximately), the associated interrupt service routine evaluates the current state of the process and decides what to do next. The next state simultaneously toggles P1V and P2V and Timer 1 period is set to 25µs. After that time, the core sets P1H high and after 35µs, it toggles P1V and P2V again. These steps reproduce the first part of the readout period corresponding to vertical register shifting in one position. Exact timer calibration is required in order to achieve a specific output pulse width.

The next step includes the generation of horizontal register phase signals needed to retrieve output voltages from each charged pixel. The step is the most critical because the core has to generate control signals, synchronize ADC modules and properly store data. Signals P1H and P2H are assigned to OC modules configured in toggling mode. Thus, each time Timer 2 reaches the configured period, the signals change from its past

state. P1H starts high (from past state), P2H starts low and Timer 2 period is configured to achieve 600ns pulse width; the two signals share the same timer to guarantee synchronism. RG pulse is also created by means of an OC module in PWM mode.

Timer 1 controls the number of P1H, P2H and RG pulses to be generated. The required period to generate a specific number of pulses N_p is given by $12N_p - 6$. The expression allows computing the total amount of time required for generating pulses with 1.2µs period. Note the expression includes -6 as the correction factor required to generate the last pulses according to the established timing diagram. The total amount of time is then divided by the period of the timer clock (50ns) to determine the register value. The CCD sensor employed requires the generation of 1508 P1H and P2H pulses, corresponding to horizontal dimension in pixels. After the required amount of time has passed, Timer 1 interrupt service routine stops the timers controlling OC modules, ADC module and DMA module. The process is repeated 1001 times to achieve extraction of 1508x1001 pixels in the CCD device.

Note these procedures occur while information from the sensor is delivered serially and synchronously. Hence, the core must also manage USB connection control, ADC module functionality and data transfer to memory.

Although PIC32 contains a 10-bit ADC, we retrieve only one byte per sample, to provide 256 intensity levels, enough for gray-scale images in the application. CCD outputs data according to P1H, P2H and RG periods. We decided to synchronize ADC conversion period with RG signal generation. Acquisition time is set to 150 ns, the minimum manufacturer requirement. Remaining time for each conversion is used in the digital conversion process. When Timer 3 reaches its configured period, acquisition stops and the module starts the conversion process, placing digital outputs always in the first position of ADC Buffer Registers.

Using DMA module features, data transfer occurs almost independently from normal flow of the core main routine. Figure 3 shows a virtual connection diagram. When configured in the most basic mode, a DMA module functions in the following manner: after activation of the associated IRQ, DMA module transfers a number of bytes configured as cell size; the process repeats each time the IRQ activates until moved data size matches the configured block size. Source and destination memory addresses must be set before operation starts. The module autonomously handles memory pointer increment.

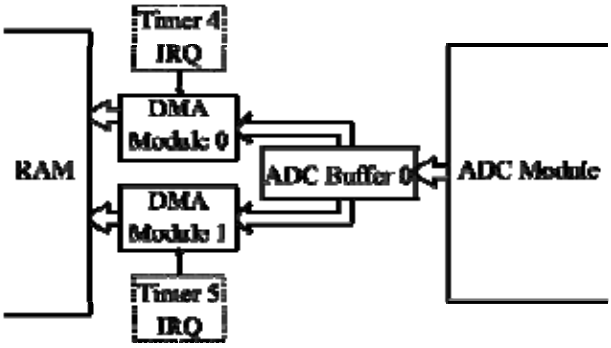

Figure 3. Connection diagram between blocks involved in ADC data retrieval.

Embedded Timers 4 and 5 control data movement between ADC buffer and RAM. The core configures the timers to have twice the period of P1H and P2H. One of the timers is “in phase” with the signals, while the other is shifted by a factor of half its own period, creating an alternating effect in generating IRQ coming from the timers. The purpose of this scheme is to provide more time between one DMA transfer and the next one. The solution considered that ADC buffer is not always available, since it is flagged as busy during conversion time. If only one DMA module is used, the result would be data loss caused by memory access conflicts. Figure 4 shows alternation of IRQ generated by the timers. Both DMA modules are configured with the same source address correspondent to the location in memory of the first ADC buffer register. Each DMA module handles 754 bytes (half-line length).

State 6 shown in Figure 2 corresponds to DMA configuration for each received line from the CCD sensor. Each time a line is retrieved, the core establishes whether the current buffer is full or not. If the current buffer is not full, interrupt routine modifies DMA destination address to point to the proper memory location to store the next 1508 bytes of data. If the buffer is full, the interrupt routine checks if the second buffer is available and not being used in USB transfers; in that case, the DMA module is configured to redirect data from ADC buffer to the first position in the new buffer.

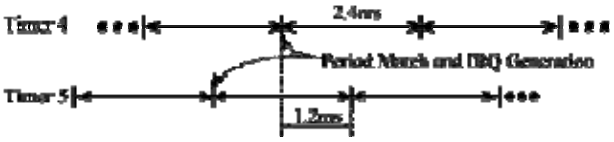

Figure 4. Time diagram of Timer 4 and 5: period match and IRQ generation.

After data storage, the core must guarantee that information is transferred to the computer. Therefore, the core distributes USB management tasks to take advantage of available bandwidth. USB protocol 2.0 allows for different transfer rates (Low Speed, Full Speed, High Speed), and four possible transfer types: control, interrupt, bulk and synchronous. For more details, the reader is referred to [12] and [13]. The most suitable transfer type for the designed acquisition system is bulk, because there is not a transfer rate requirement and large blocks of data are being transferred to the computer.

USB implementation was based on Microchip USB Stack Framework [14]. The stack provides most register and memory configuration routines in order to successfully establish a USB connection. The stack uses common C directives for easy configuration of parameters like number and size of endpoints, associated devices descriptors, etc. The stack also embeds necessary functions, handlers, and routines to initialize the USB module, endpoints and to handle USB associated events. For the particular application, two data endpoints were selected to communicate with the computer. Determination of number of endpoints and their size is based in transfer tests made under different conditions. Results are shown in the following section. The selected size of the endpoint corresponds to the maximum allowed size for bulk transfers, 64 bytes.

The system implements a double buffering scheme, thus data is moved to two different buffers in RAM according to current availability. A data buffer is sent using two simultaneous data endpoints; each endpoint transfers data to fill half buffer. Given that each buffer was chosen to have a size of 9048 bytes (6 image lines) and the configured endpoint size is 64 bytes, information is split into 142 packets. Consequently, there is an overload of 40 bytes per buffer, causing a total transmitted buffer size of 9088 bytes. Thus, overhead causes 0.44% increment in the overall total information size.

Figure 5 shows a diagram with the main flow of the control scheme. The flow handles high-level communication between the core and the computer. Three main events can occur: the computer may command the microcontroller to initiate proper signaling to the CCD sensor; the computer may poll an available buffer to retrieve

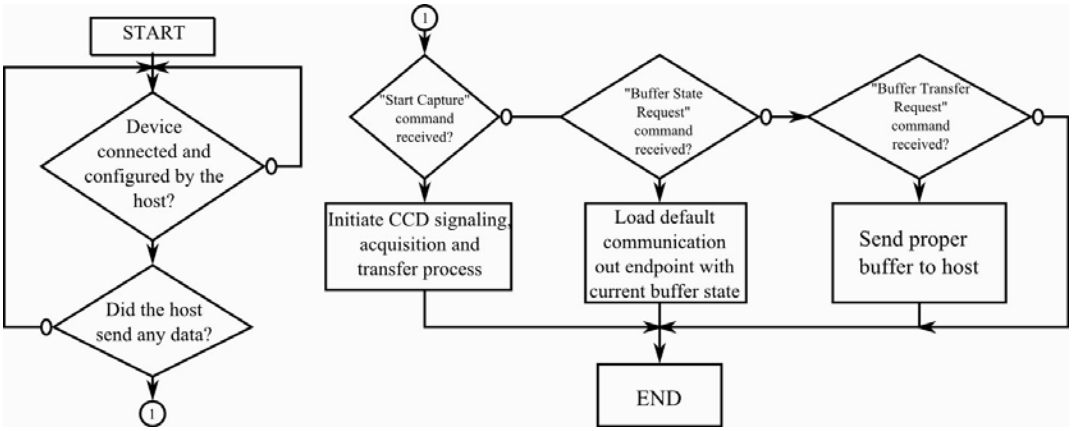

Figure 5. Flowchart of the main routine for the designed system.

information; if the core answers positively to the previous request, the computer may command it to send the contents of the corresponding data buffer.

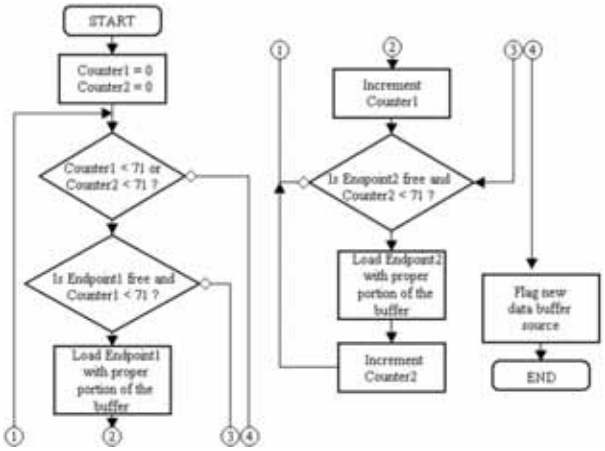

Figure 6. Diagram illustrating the USB transfer using two simultaneous endpoints.

During image acquisition, the last two events are iterative: the computer is always polling for an available buffer and after a positive answer it requests data buffer from the core. The process repeats until all 167 data buffers that build up the image are sent.

Figure 6 shows the process for transferring one buffer filled with data. The process starts once the core confirms that data is available to be transferred. As mentioned before, 9088 bytes are sent by means of two endpoints working simultaneously. This subprocess is controlled with two independent counters initially set to zero, one for each endpoint. The core then determines if the first endpoint is free and if there are data to send. If the conditions are satisfied, the core loads the first endpoint with a portion of 64 bytes of the first 4544 bytes in the data buffer, causing an increment of the first counter by a unit. Afterwards, the device will do the same process with the second endpoint but transferring the last 4544 bytes in packets of 64 bytes. The process repeats until all 142 packets are sent.

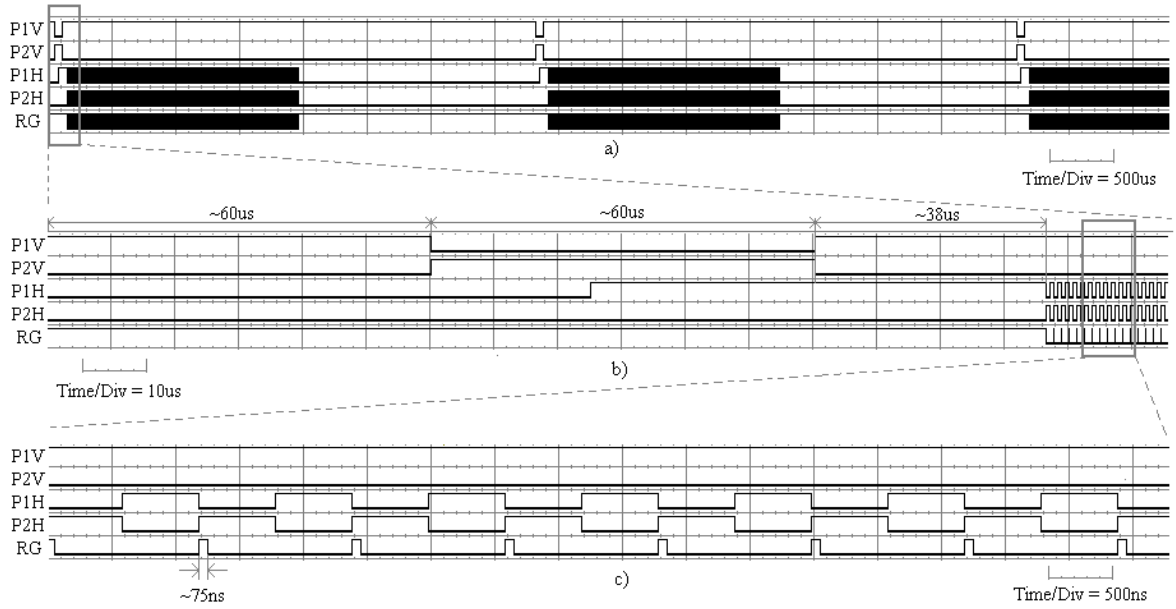

Figure 7. a) Magnified view of signals for retrieving only three of the 1001 lines in the sensor.
b) Magnified view of the initial process for retrieving one line.
c) Magnified view of the horizontal shift register phase signals and the reset gate signal when active.

4. Results

Tests were made in order to measure performance of both the signal generation scheme and the USB transfer protocol. Experiments employed a logic state analyzer to observe if PIC32 generated required control signals. Figure 7 shows these signals meeting timing requirements, hence demonstrating that the system is capable of controlling the CCD sensor.

USB transfer tests were made in order to determine the appropriate combination between the number of endpoints and their size, considering three factors: endpoint size, number of endpoints and simultaneous USB transfers with image transfer. For each factor, the following levels were considered: endpoint sizes of 16, 32 and 64 bytes; one, two or three simultaneous data endpoints; and common simultaneous transfers consisting in a 3GB file transfer to a USB storage device, an active document (1200 pages) printing process and no other external connection causing additional bandwidth consumption. Tests employed full factorial experiment, considering every possible combination. Each combination was repeated 10 times, as the maximum observed variability for all combinations was only of 4%. The null hypothesis is: “changes in factors do not interfere in total transfer time for a single image”. Figure 8 shows a summary of the results obtained. For the results in Figure 8a, the maximum standard deviation was 43ms; as seen in the range of values, no overlap is possible between combinations.

4. Conclusions

Results shown in Figure 8a prove that for all number of endpoints configured, the best results correspond to endpoints with maximum allowed size for bulk transfers because the core sends more data within one time frame in the bus. The number of endpoints also showed to be important in total transfer time variation.

Results allow rejection of the proposed null hypothesis, as the number of configured endpoints and their size influence total transfer time. Although the best combination was proved to be three endpoints with 64-byte size, we implemented a data transfer with two endpoints as this scheme facilitates data management. The difference between transfer time with two and three end points is small. However, when comparing connections with one end point, transfer time is 50% higher than time used with two and three end points.

Figure 8b shows the total transfer time with a 95% confidence interval for two endpoints with 64-byte size, thus results prove that the total transfer time is statistically independent of simultaneous transfers for the implemented configuration. The situation may be explained considering that experiments used a transfer rate of 12 Mbps over a bus with higher bandwidth of 480 Mbps, which means that sufficient bandwidth is available for common simultaneous transactions.

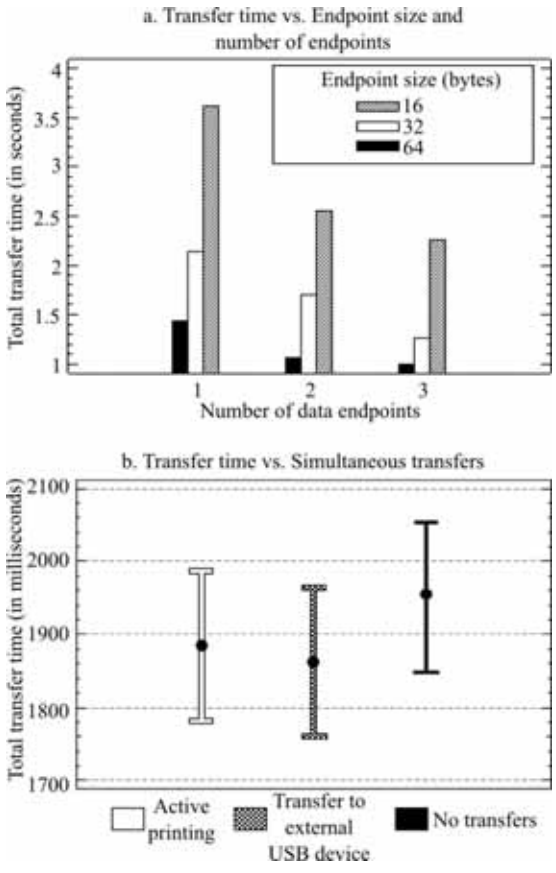

Figure 8. a) Total transfer time for different number of data endpoints and endpoint size.
b) Total transfer time for different types of simultaneous transfers.

References

[1] Bernd Jähne, Practical Handbook on Image Processing for Scientific and Technical Applications (C-5), Second Edition, CRC Press 2004, pp. 169–206.

[2] Characteristics and use of FFT-CCD area image sensor, Hamamatsu Solid State Division, Hamamatsu Photonics, Japan, 2003, September.

[3] Universal Serial Bus Specification, Revision 2.0, USB Implementers Forum, 2000, USA, April.

[4] D. Anderson, USB System Architecture (USB 2.0), 1st Ed, Addison-Wesley Developer's Press, 2001, pp. 11-209.

[5] Hong S., Jung H., Kim K., So S., Kim J., Yoo S., Yoo H. & Kim H. , Development and evaluation of a CMOS sensor-based digital intra-oral radiographic system, IEEE Transactions on Nuclear Science, vol.52, no.1, pp. 256- 261, Feb. 2005.

[6] Bin Y. & Liqiang W., A CMOS Based Image Acquisition System for Electronic Endoscope, Symposium on Photonics and Optoelectronics, SOPO 2009, pp.1-3, 14-16 Aug. 2009.

[7] Proffitt J., Hammond W., Majewski S., Popov V., Raylman R.R. & Weisenberger A.G., Implementation of a High-Rate USB Data Acquisition System for PET and SPECT Imaging, IEEE Nuclear Science Symposium Conference Record, 2006., vol.5, pp.3063-3067, Oct. 29 2006 - Nov. 1 2006.

[8] Park J., Lee D., Kim D. & Jeon J., Implementation of a stand-alone color image transfer circuit using the USB 2.0 interface, ICCAS '07 International Conference on Control, Automation and Systems, 2007, pp.2323-2328, 17-20 Oct. 2007.

[9] Chen C., Chen P. & Liao T., The graphic system design in color image capturing system using the USB interface, 2010 International Symposium on Computer Communication Control and Automation (3CA), vol.1, pp.119-121, 5-7 May 2010.

[10] Tian J., Gao M., Dong N. & Xu J., Image Data Acquisition and Transmission System Based on ARM, International Conference on MultiMedia and Information Technology, 2008. MMIT '08., pp.361-364, 30-31 Dec. 2008.

[11] Gao H., Yuan L. & Wang T., The design of a network video transmission system based on DSP and USB, 2010 2nd International Asia Conference on Informatics in Control, Automation and Robotics (CAR), vol.1, pp.121-123, 6-7 March 2010.

[12] S8980 Datasheet, CCD area image sensors, Hamamatsu Solid State Division, Hamamatsu Photonics, Japan, 2011, May.

[13] PIC32MX3XX/4XX Datasheet, Revision G, Microchip Technology Inc., 2010, April.

[14] USB Device Stack for PIC32 Programmer's Guide, AN1176, Revision A, Microchip Technology Inc., USA, 2008, February.