Design of Vision Embedded Platform with AVR

In-Kyu Jang, Dai-Tchul Moon, Hyoung-Kie Yoon, Jae-Min Jang, Jeong-Seop Seo

1Dept. of Information & Communication Engineering, Hoseo University, Republic of Korea, ikjang86@gmail.com, dtmoon@hoseo.edu, yhk4618@nate.com, isunset@me.com, sjs1066@nate.com

Abstract

In this paper, we designed motion detection system based on AVR ATmega2560 micro controller unit(MCU). It is a processing platform for high performance vision sensing using a low specification processor. In the platform, a vision processing system is implemented in which motion detection is performed by comparing vision data from camera module. It also includes a power saving switch module that can realize fancy operations by commands resulted from analysis of images in the embedded vision system and delivered through serial communications. Though a vision embedded system using 32bit processor based on OpenRISC was implemented in our previous work, we tried to make system circuits simple and realize the system with lower cost using AVR processor at this time, and accomplished more sensible platform. This system is simulated and verified by completed test with debugging and implemented in the form of PCB.

Keywords: vision; embedded system; AVR;

1. Introduction

Using vision data technology is being wider application area from intelligent robot to mobile machine like cell-phone. So, it increases to demand methods available high performance vision processing with being satisfied with constraints on embedded system.

Following industrial development, in case of Vision sensing, It is a green IT technology available to apply for decreasing increased power consumption to shut down streetlight, electronic products etcetera automatically in unnecessary condition and time.

In over previous work, we already designed Energy Saving Vision Embedded system based on OpenRISC and AVR ATmega128 8bit micro-controller. It was simple circuit and got a profit low cost than OpenRISC. ATmega128 doesn’t have enough memory space and I/O pin. So, we compared pixel data 54*40 of 320*240(vertical*horizontal).

In this paper, a platform is vision embedded system based on AVR processing image data inputs from camera. Image data inputted through a camera is processed by a vision algorithm and the data is saved into external memory and the data is displayed through graphic-LCD and the processed result is transmitted through hand-shaking protocol communication to the application.

2. Overall platform

2.1. System architecture & operations

This system is consists of three parts, which are camera module for getting an image data, graphic LCD module for display and processor for process the image data. The image data from camera module is saved into FIFO and each pixel data is transferred to processor progressively following synchronize signal from processor.
And the processor sends the image data to Graphic LCD and the Graphic LCD display the real time image data. Size of real time image data is 320 by 240 pixels.

At this time, software is designed with C-language and the designed program is compiled with AVRStudio. After compiling, hex file is being created and downloaded to processor.

![Overall Architecture](image)

**Figure 1.** Overall Architecture

### 2.2. Processor and memory

AVR is included low power 8bit Microprocessor, 32*8 general purpose register, GPIO, Memory-Flash Memory, SRAM, EEPROM. It is able to do USART and TWI serial communication. There are SPI for In-System Programing and JTAG for programing and debugging.

In past paper, we used AVR ATmega128 micro controller. Atmega2560 has more GPIO lines than ATmega128. So, we substitute ATmega128 to ATmega2560 in this paper.

When we used ATmega128, it doesn’t have any spare GPIO lines. So, we can’t connect any more interfaces directly on MCU. But, through change of processor 128 to 2560, we can connect additional memory interface for save much more pixel data.

In case of AVR, it can be connected external memory up to 64K Bytes. So, we connected external SRAM. Size of real-time image is 320 by 240 pixels. The image data is saved into FIFO and each pixel data is transferred progressively to external SRAM through MCU.

In the previous work, when we used Internal SRAM, it was able to compare 2160 pixels of 1 frame. For using external memory, we can compare more pixels 8 times than previous work.

Case of connecting external SRAM, through table 2, we can show that rate of comparing pixel is more sensitive than using internal SRAM.
TABLE 1. Comparing AVR ATmega128 & ATmega2560

<table>
<thead>
<tr>
<th>Feature</th>
<th>ATmega128</th>
<th>ATmega2560</th>
</tr>
</thead>
<tbody>
<tr>
<td>MIPS</td>
<td>16MIPS at 16MHz</td>
<td>16MIPS at 16MHz</td>
</tr>
<tr>
<td>FLASH</td>
<td>128K</td>
<td>128K</td>
</tr>
<tr>
<td>EEPROM</td>
<td>4K</td>
<td>8K</td>
</tr>
<tr>
<td>SRAM</td>
<td>4K</td>
<td>Able</td>
</tr>
<tr>
<td>I/O Lines</td>
<td>53</td>
<td>86</td>
</tr>
<tr>
<td>Timer/Counter</td>
<td>PWM, ADC, SPI Serial Interface, TWI, Interrupt etc</td>
<td>PWM, ADC, SPI Serial Interface, TWI, Interrupt etc</td>
</tr>
</tbody>
</table>

**Figure 2.** External SRAM Connected to the AVR

TABLE 2. Case of Internal and External SRAM

<table>
<thead>
<tr>
<th></th>
<th>Internal SRAM</th>
<th>External SRAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sum of comparing pixels</td>
<td>2160 (54 * 40)</td>
<td>19200 (160 * 120)</td>
</tr>
<tr>
<td>Rate</td>
<td>1</td>
<td>8.89</td>
</tr>
</tbody>
</table>

2.3. Hand-Shaking Protocol

A handshaking protocol is a method by which data is estimated to be received, the content of the primary information sent and the parameters of the response. A handshake can be a single-query-response process, or it can be many processes.
When MCU transfers the compared results to slave, we used hand-shaking protocol for serial communication. Master and Slave are connected 3 wires – Ack signal and Req signal for informing communication state, connection for transferring data – each other.

Master provides Req signal informing data’s ready. Slave provides Ack signal getting data correct.

- Master provides Req signal informing data’s ready. Slave provides Ack signal getting data correct.
- Master prepares data and output Req=’1’. If Ack=’0’ on Slave, it means that receiving is not finished yet. So, Master maintains the state.
- Becomes Ack=’1’ on Slave, it means correctly data transferred. Therefore, Master outputs req=’0’. If Ack=’1’ on Slave, it means that receiving is not finished yet. So, Master maintains the state.
- If slave outputs Ack=’0’, transferring data is finished.

![Handshaking protocol](image)

**Figure 3.** Handshaking protocol

### 2.4. Pixel Comparing

In order to detect motion in each partitioned region, image comparing program is developed and compiled C code is downloaded into EEPROM of vision embedded platform. For image comparing, neighbor frame is required. The critical issue is method of image saving and comparing

- Get image data from camera and save the image data at FIFO
- Save the image data progressively at external memory from FIFO following vertical synchronization and horizontal synchronization
- Get next image data from FIFO and compare corresponding pixel data. If the corresponding pixel data are different, MCU transfers the result and instruction to application through hand shaking protocol
3. Conclusion

In this paper, vision embedded platform for energy saving has been implemented, making easy development and using it. Hardware part of this system is implemented with 8bit microcontroller. Software part accomplishes image comparing algorithm from getting image by camera by C language.

This embedded platform gets images from camera, detects motions in partitioned image regions, and sends results to switching circuit. LED turns off for still state and turns on by the detection of movement in video camera. This operation means automatic energy saving operation simulated by vision sensor.

Platform is designed C language based on AVR ATmega2560 to available application in various field. Through Designing additional application, also, developer can modify software fluidly that user want.

REFERENCES


