# Development of FPGA-based Programmable Timing Controller

Soung Moon Cho* and Jae Wook Jeon**

* School of Information and Communication Engineering, Sungkyunkwan University, Suwon, Korea
(Tel : +82-31-290-7237 ; E-mail: ampmoon@ece.skku.ac.kr)
** School of Information and Communication Engineering, Sungkyunkwan University, Suwon, Korea
(Tel : +82-31-290-7231 ; E-mail: jwjeon@yurim.skku.ac.kr)

**Abstract**: The overall size of electronic product is becoming small according to development of technology. Accordingly it is difficult to inspect these small components by human eyes. So, an automation system for inspecting them has been used. The existing system put microprocessor or Programmable Logic Controller (PLC) use. The structure of microprocessor-based controller and PLC use basically composed of memory devices such as ROM, RAM and I/O ports. Accordingly, the system is not only becomes complicated and enlarged but also higher price.
In this paper, we implement FPGA-based One-chip Programmable Timing Controller for Inspecting Small components to resolve above problems and design the high performance controller by using VHDL. With fast development, the FPGA of high capacity that can have memory and PLL have been introduced. By using the high-capacity FPGA, the peripherals of the existent controller, such as memory, I/O ports can be implemented in one FPGA. By doing this, because the complicated system can be simplified, the noise and power dissipation problems can be minimized and it can have the advantage in price.
Since the proposed controller is organized to have internal register, counter, and software routines for generating timing signals, users do not have to problem the details about timing signals and need to only send some values about an inspection system through an RS232C port. By selecting theses values appropriate for a given inspection system, desired timing signals can be generated.

**Keywords:** Timing Controller, FPGA, One-chip

## 1. INTRODUCTION

The overall size of electronic product is becoming small according to development of technology. It is difficult to inspect these small components by using only human eyes. Therefore, the development of controller that is able to inspect small electronic product is needed. Generally, the former controller based on microprocessor and PLC (Programmable Logic Controller). These controllers need the peripheral circuits like the memory (ROM or RAM) and I/O port in order to operate microprocessor, DPRAM in order to communicate with Host PC, Feedback Counter in order to get current position information of motor. Therefore, These Controllers have several problems that are overall system complexity, large size and space, much cabling, large power consumption, and additional high costs. Also, it is difficult that the microprocessor and PLC process many tasks according to high-speed encoder signal at the same time because they store data in RAM then process it in ALU (Arithmetic and Logic Unit). So, these controllers cannot control many motors because of control by using software, communication with Host PC, and input of high-speed encoder signal.

This paper proposed the development of FPGA-based programmable Timing Controller to solve above-mentioned problems. This proposed controller is designed with a FPGA by using the VHDL (Very high speed integrated circuit Hardware Description Language) that is language for design hardware. Recently, the high-performance FPGA that have many logic gates and high-speed is developed. And it is very chip, the embodiment of circuit in FPGA have some merits like overall system simplicity, low power consumption, and low cost.

The proposed controller consists of main control part to control overall system, sensor input part to get external sensor signals, feedback counter to get current position information of motor and interface part to communicate with Host PC. Main control part consists of memory part that converts difference of distance between part and part into value of encoder signal and stores, and compares part that compares current point with operating point.

The contents of this paper are as follows. In the section 2, explain about overall system. In the section 3, explain about structure and function of proposed controller. In the section 4, experiment with proposed programmable Timing Controller. Finally, the conclusion is described in the section 5.

## 2. EXPLANATION OF OVERALL SYSTEM

The Programmable Timing Controller is controller that is used for system to inspect small electric products. It is as following that need to inspect small electric product. Part feeder is needed to align electric product. And a conveyor belt is needed to move aligned product from part feeder to a detector. Also, sensor is used to recognize product, it accepts via conveyor-belt-output-value. The distance between moving electric products can know using input of sensor and encoder pulse, and it offers distance that electric products moves from sensor to vision controller.
The Fig. 1 is the architecture of system for inspects small electric products, above-mentioned. The system paired off sensor and actuator. The pair is basic of system-action. Let see inspection order, a feeder aligns small electric products and supplies them to a belt for being inspected. Small electric products are transferred along a belt driven by an electrical motor. An encoder is attached with the electrical motor to provide its pulse as a master signal. The supplied part to belt start on moving. According to priority, the moving-parts are sensed in Sensor_0. And the parts have blown out of belt in Valve_0 if the space between parts is so near to process data in camera. The parts that keep suitable distance to photograph in camera are sensed in Sensor_1, and the parts photographed in camera. Next of the photographing, the parts are sensed in Sensor_2, once more. And the parts are handled separating by each extraneous matter, bad product and good product in valve 2,3,4. The extraneous matters are things out of inspection and dust.

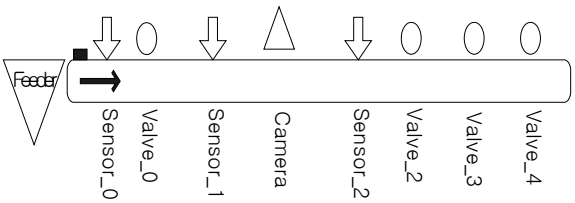



Fig. 1 An inspection system for small components

## 3. DESIGN OF TIMING CONTROLLER BY USING FPGA

The Fig. 2 is a block diagram of the proposed Timing Controller designed with a FPGA by using VHDL. The internal architecture of FPGA consists of host interface part, main control part, sensor input part and encoder counter part. The host interface part communicates with a PC by RS-232C. The main control part controls the overall system. The sensor input part inspects sensor input from external devices, and transfers the sensor data to main control part. The encoder counter part counts encoder signal from electric motor, and finds a direction and position information. As know in block diagram, overall system is in one-chip because all circuit designed in FPGA.

Additionally, let's see some signals that are inputted from FPGA outside. The sensor signal is signal to detect external product. The encoder signal is output signal from electric motor. The reset signal is signal to initialize timing controller. The CLK is standard clock for act timing controller. And the frequency of CLK is 24.576MHz. The reason that uses 24.576MHz is to use in Serial communication. As see in equation (1), 9600 divides 24.576MHz by 2560. The 9600 is baud-rate of serial communication.

$$9600 = \frac{24.576MHz}{2560} \qquad (1)$$

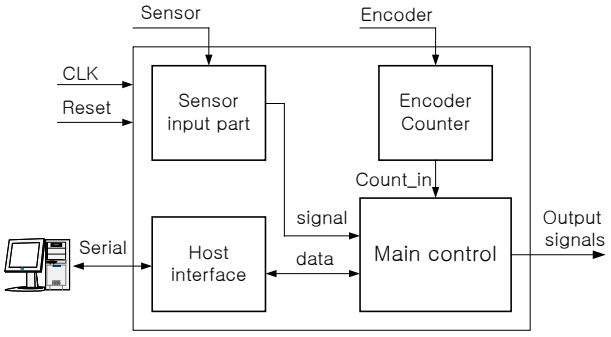



Fig. 2 Block diagram of the proposed Timing Controller

Design of all systems uses IEEE standard VHDL. So, we can implement system that wants to change of VHDL program in case a change of system need. In this system, the FPGA device is used XC2S200-PQ208-5 that is Spartan series in Xilinx Inc. [1], [2]. It offers enough resource that is capacity of 200,000 gates and 140 I/O. The resource is enough in implementation of system. Design Tool is Xilinx ISE 4.2i.

### 3.1 Sensor input part

Sensor input part inspects sensor input from external devices. This part stores encoder pulse from sensor to actuator at the main controller, as sense input signal by the sensor.

The Fig. 3 is method for recognition of sensor. The valid sensor signal recognized that sensor input signal maintain High while five-times at synchronous with basic CLK (24.576MHz). The reason is to protect that the noise recognized sensor input. The sensor input part decides High, if signal is High for every time. Otherwise, the sensor signal decides noise, and disregarded.

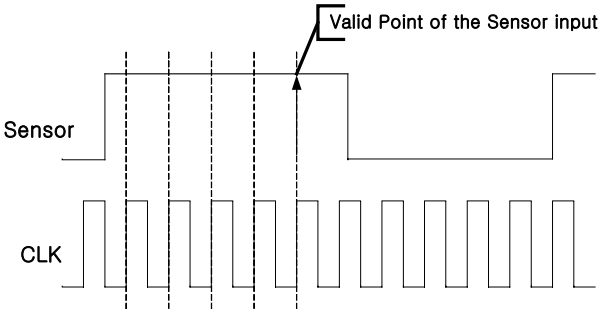



Fig. 3 Sensor input point

### 3.2 Encoder counter

The Fig. 4 is method for recognition of encoder signal. The recognition of encoder signal resembles that the recognition of sensor input in method. As the recognized encoder pulse, the valid encoder pulse signal recognized that encoder pulse signal maintain High while three-times because the encoder pulse signal is shorter than sensor input signal. That is, the valid encoder pulse signal recognized a High if the High signal maintained at three-time, and recognized a Low if the Low signal maintained at three-time. The encoder signals, generated above method, are used by basic-clock of main control part, by new encoder signal. When detect encoder signal, the method is to detect encoder signal's edge, too. But in this circuit selected method like Fig. 4. It is because that the encoder signals of motor have the many noise. And timing controller uses forward signals only.

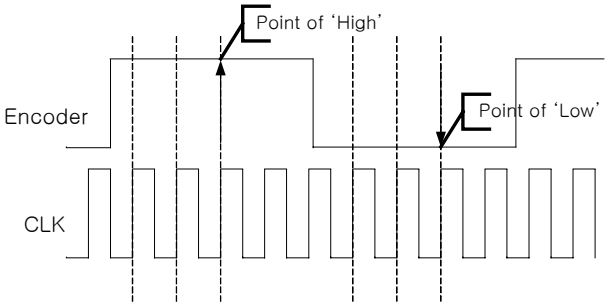



Fig. 4 Encoder input point

### 3.3 Main control part

Fig. 5 is the architecture of main control part. The main control part consists of 2 counters, memory register part, and comparator and drive part. The counter counts encoder clock - made from encoder counter part. The Memory register part stores a value of counter 1 every-time sensor input. The Comparator part compare between memorized value at the memory part and the value of counter 2, and generate drive-signal if the two values is equal. The input consists of encoder CLK, Sensor input signal and Serial input. The output consists of drive-signal and serial-output. The serial input is value that is inputted from PC, and it is distance value from

sensor to actuator. The serial output sends to PC that the distance between electric product, and the action signal of driver.

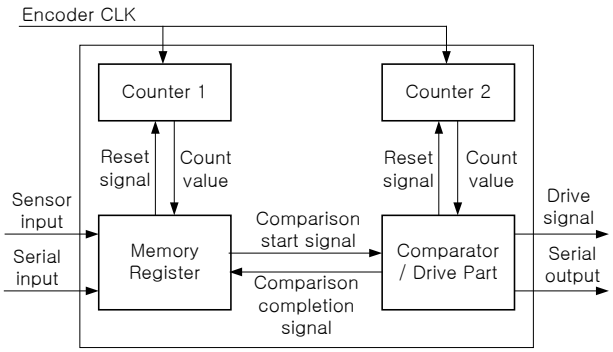



Fig. 5 The architecture of Main Control part

The table 1 appearance registers of proposed controller.

Table 1 Registers in the proposed Controller

| Name | Length | Functions |
|---|---|---|
| En_clk | 1 bit | The encoder clock from encoder counter part. |
| Start_bit | 1 bit | Inform that first sensor value entered. |
| Read_bit | 1 bit | Mark that read value as the sensor entered. |
| Sen_high | 1 bit | Inform that sensor input entered. |
| Action_bit | 1 bit | Inform that product reached to actuator. |
| ac_1_bit ~ ac_8_bit | 1 bit | Inform whether compare with value that is stored to any register. |
| s_count_bit | 1 bit | Keep the actuator's action |
| Memory_1 ~ Memory_8 | 20 bit | The store-buffer of distance from sensor to actuator. |
| First_value | 20 bit | The input buffer distance from sensor to actuator entered, by PC. |
| Memory_ch | 8 bit | The register that inform whether store value to any memory buffer at sensor entry. |
| Action_ch | 8 bit | The register that inform whether compare with any memory buffer. |
| Count1 | 20 bit | The counter to calculates distance from sensor to actuator. |
| Count2 | 20 bit | The counter to compare whether product reached to actuator. |
| Count3 | 20 bit | The counter for keeps actuator's action while some time. |

### 3.3.1 Counter part

Counter part is 20bit up counter that increase via encoder clock. The counter size declared 20bit for use mediocrity controller because every systems have other encoder resolution. This controller used two counters. The counter 1 operates for Reset signal by memory register part, and store to memory register part at the every sensor signal. The counter 2 is counter that use in comparator part. The memory register part sends compare-start-command to comparator part, then comparator part sends count-start-command to counter 2. The counter 2 is reset by command of the comparator at the electric product arrived to actuator.

### 3.3.2 Memory Register part

The memory register part is part that store encoder counter value whenever sensor input enters. The buffer to store counter value used 8. As the first sensor input, the first buffer stores distance from sensor to actuator stored by serial communication. From the second sensor input, the buffer stores encoder pulse value from front sensor inspection to next sensor inspection. Then, if the distance sensor to sensor bigger than first value - distance value from sensor to actuator by serial communication entered -, the buffer stores first value. If do not like above and stores distance sensor to sensor, the product passed actuator and the actuator operates without product. In other words, if there is no sensor input when counter value passed over first value, stop the counter's action. And the memory buffer wait for sensor input. As the sensor entered, the memory buffer stores first value to next buffer and reset counter. For above operation, memory register part need register for know order of memory buffer that value is stored and need comparator that decide whether counter value is bigger than first value. If all buffers are full, store value to first register when next sensor entered.

The Fig. 6 is flowchart of memory register part designed by FPGA. Let see the order. Priority, controller wait for inputted first value of PC by serial communication. Then PC sends distance from sensor to actuator and signal for operating by controller. Controller waits sensor input after operating command, and if there is sensor input, whether is first sensor input. If the first sensor input, controller stores first value and operate the counter, and rotate the register for inform whether buffer that is stored at next time is anything. From second sensor input, stores counter value from previous sensor input. And send command for comparator operating to comparator part. Like above order, memory register part stores counter value, reset counter, and sends command to comparator part every sensor input.

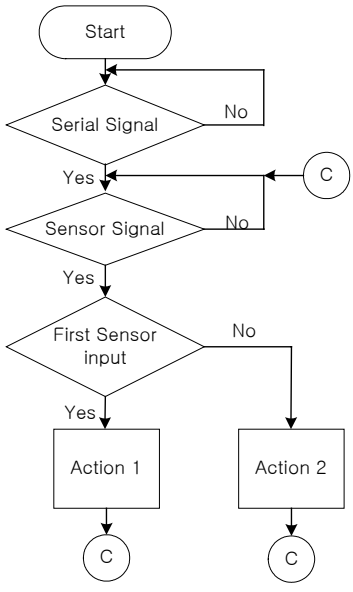



Fig. 6 The flowchart of Memory Register part

### 3.3.3 Comparator / Drive part

The Fig. 7 is flowchart of comparator and drive part. Comparator part start operations as received command for comparison at memory register part and entered information for register to compare. Priority, this part compare first input product. These increase counter 2 and compare current count

value with value of memory buffer. If the same, it is that product reached to actuator. Then, controller sends command to actuator, reset counter, and return to compare with next memory buffer. Then another counter – counter3 – operate. This counter is counter that is kept actuator's action state during some time. Because, actuator is need the time to operate the product. The counter 3 is auto reset. And controller sends action-ending signal to PC after reset counter 3.

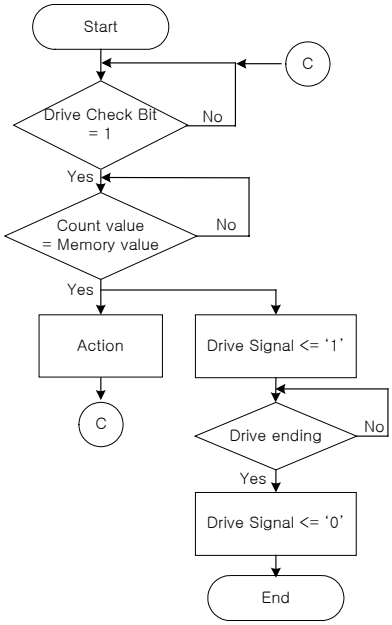



Fig. 7 The flowchart of Comparator part

### 3.4 Host interface

The controller does data input/output for data exchange with PC by bi-direct serial communication through RS232C. Serial communication is synchronism communication and asynchronous communication. This controller selected asynchronous communication.

The Fig. 8 is architecture for serial communication. This controller receives basic clock of 24.576MHz and communicates by 9600b/s. Also, data exchange in controller operates through 8-bit data bus. So, serial to parallel proposed.

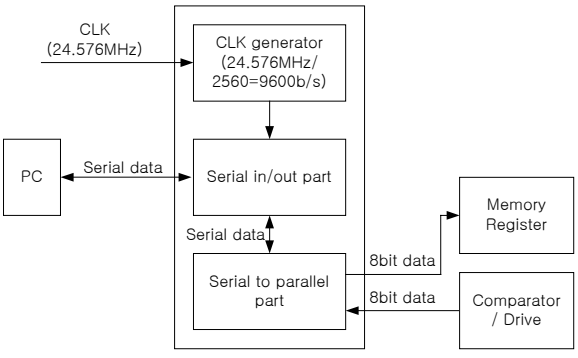



Fig. 8 The architecture of Host interface

#### 3.4.1 As the data output

When controller sends data to PC, is simple comparatively. The controller transmits data by 9600b/s, if there is sending request. The order of data transmission is start bit '0', 8bit data, and stop bit'1'.

#### 3.4.2 As the data input

It is different between receiving method by PC and sending method by PC. The Fig. 9 is method for receive serial input. If the start bit inspection interrupt operates, inspector read again after a little delay for know whether the signal is valid signal. If the signal is same, the signal recognized start-bit. If the signal is different, the signal recognized noise and ignored. And the detector wait next start-bit. If the detector recognized valid start-bit, next-timer start by 1.5-times of 1 bit input length. And the controller read data as the time-up interrupt entered. So, controller can read data at the middle of data length. And the controller read 8-bit data. After the controller read 8-bit data, confirm stop-bit. Stop-bit is not entered, the data is ignored and request by PC for receive data again

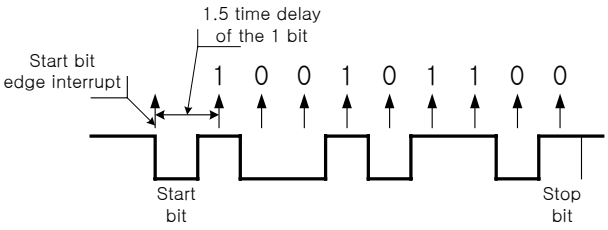



Fig. 9 The method of Serial Receive

## 4. EXPERIMENT

The Fig. 10 is the block-diagram of implemented system for experiment. Because we have not inspection system of electric products, we ourselves manufactured test system for experiment of timing controller. An explanation about the test system is behind. The input circuit and output circuit designed because voltage difference between control part and implemented test system. The control part operates by 3.3V and the implemented test system operates by 24V. The test system consists of magnetic valve and LED. And the output signals are sensor signal and encoder signal.

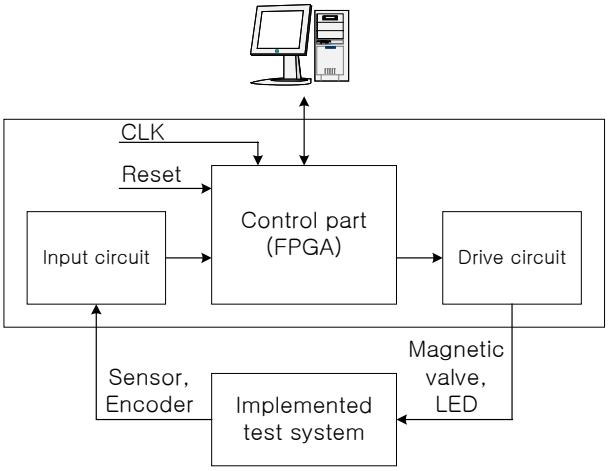



Fig. 10 The block diagram of implemented system for experiment

#### 4.1 Implemented timing controller

The Fig. 11 is appearance of implemented system. As shows in Fig. 11, developed timing controller is simplified because all functions are in the one-chip FPGA. The circuits around FPGA are circuits for operate sensor and actuator.

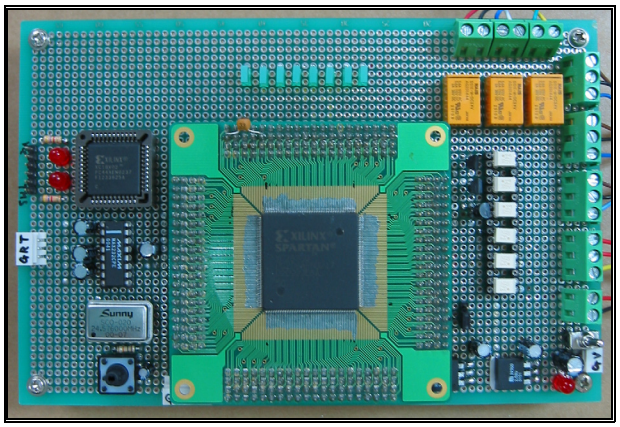


Fig. 11 Implemented Timing Controller

The FPGA device is used XC2S200-PQ208-5 that is Spartan series in Xilinx Inc. This device's capacity is 200,000 gates and this research used 16,536 Gate that is the 17% amount of them. The Fig. 12 shows FPGA's whole resources amount used.

```
Design Summary
--------------
  Number of errors:     0
  Number of warnings:   0
  Number of Slices:              1,014 out of  2,352   43%
  Number of Slices containing
    unrelated logic:               0 out of  1,014    0%
  Number of Slice Flip Flops:    783 out of  4,704   16%
  Total Number 4 input LUTs:   1,372 out of  4,704   29%
    Number used as LUTs:             1,297
    Number used as a route-thru:        75
  Number of bonded IOBs:          17 out of    140   12%
    IOB Flip Flops:                    6
  Number of GCLKs:                1 out of      4   25%
  Number of GCLKIOBs:             1 out of      4   25%
Total equivalent gate count for design:  16,536
Additional JTAG gate count for IOBs:  864
```

Fig. 12 The FPGA's whole resources amount used

### 4.2 GUI(Graphical User Interface)

The Fig. 13 is GUI that is implemented for user interface. The program development tool used Visual C++ 6.0 in Microsoft Corporation that is generalized most present. This GUI is composed of simple form for control timing controller. All functions of system can try test by simple button manufacturing. This GUI enters user's input and transmits to timing controller by serial port. The GUI consists of ROGRAM status, Transmission and Receiving.

The each function is as following. The first, PROGRAM status window is the part that shows message for user's command and message to send timing controller. The user can know result of command operating and can see received value by timing controller, by real time.

The second, transmission window inputs information to sends from PC to timing controller. Controller 1, controller 2 and controller 3 are block of interior by controller. Basically, the GUI can control 3-block in timing controller. Each value is that convert distance from sensor to actuator by encoder value. And this value transmitted as push the Transmit button. After, push Program start button. Then PC transmits start-command to controller. The timing controller is start operating after this command.

The third, the receiving window display real moving value from sensor to actuator, after timing controller operates actuator. At this window, the user can decides that inspect whether actuator's action consisted properly.

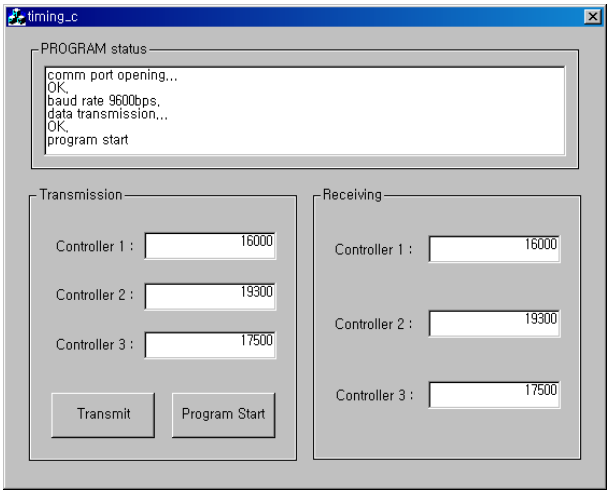



Fig. 13 GUI of Timing Controller

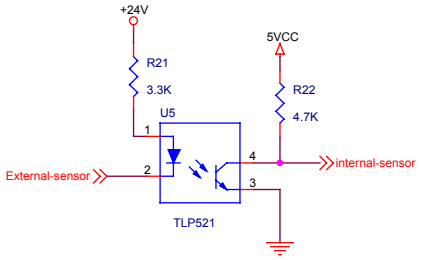
The table 2 is function table of GUI for interface with PC.

Table 2 The functions of GUI

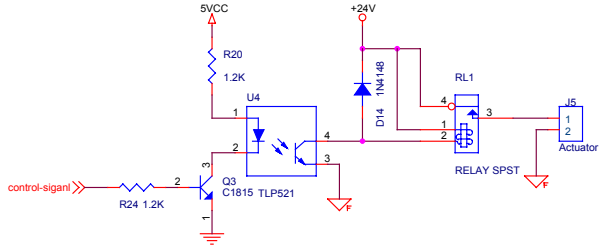
| Sections | | Functions |
|---|---|---|
| PROGAM Status | | Display state of the current communication |
| Trans -mission | Controller 1 | Input first-value for transmission to Controller 1 |
| | Controller 2 | Input first-value for transmission to Controller 2 |
| | Controller 3 | Input first-value for transmission to Controller 3 |
| | Transmit | Transmit first-value to each controller |
| | Program Start | Transmit operating start command to timing controller |
| Receiving | Controller 1 | Display receiving value by controller 1 |
| | Controller 2 | Display receiving value by controller 2 |
| | Controller 3 | Display receiving value by controller 3 |

### 4.3 The circuit of FPGA around

External device acts by 24V, like sensor and actuator. So, The system needs the circuit like Fig. 14 (a), (b). The internal and external are divided by photo-coupler.




(a) Sensor input circuit




(b) Drive output circuit

Fig. 14 The circuit to interface by external device

### 4.4 The system to test timing controller

The Fig. 15 is system to test controller. The system consists of 3 sensors, 2 LED (light emitting diode) and 1 magnetic valve. The first LED is display good product. The second LED is display bad product. And the magnetic valve do like that good product passed and bad product has blown outside.

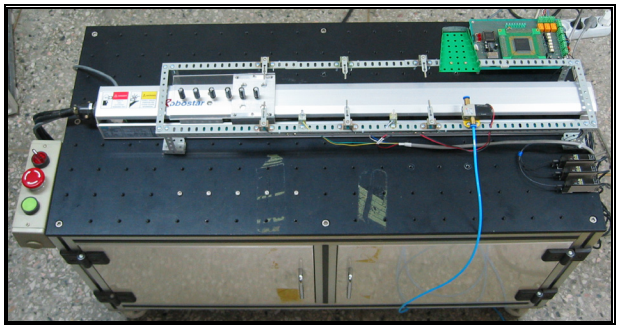


Fig. 15 Implemented test system

## 5. CONCLUSION

In this paper, we developed the programmable timing controller for inspection of small electric products based on FPGA by using the VHDL, and experimented with the test system. By the most functions of the former controller designed in FPGA, several problems caused by the complicated structure of the former controller could be resolved and improved. Since the proposed controller is organized to have internal registers, counters, and routines for generating timing signals, user need to only send some values about an inspection system through a RS232C port to generate desired timing signals.

Current, this programmable timing controller communicates with PC by RS232C. As the proceeding section, develop by controller that can communicate with other means of communication, like USB (Universal Serial Bus) or PCI (Peripheral Component Interconnect). And make GUI that has various functions to improve general purpose.

## REFERENCES

[1] *The Programmable Logic Data Book,* Xilinx Inc., USA, 1996.

[2] *Spartan-II 2.5V FPGA Family: Function Description,* Xilinx Inc., 2001.

[3] Edward O. Thorp, “The Invention of the First Wearable Computer,” *IEEE The Second Int. Symp. On wearable Computer*, PP. 4-8, Oct. 1998.

[4] LC880 User Guide, LABSmith, 2001.

[5] PulseBlaster, SpinCore Technologies, Inc. 2002.

[6] Universal Programmable Control, Electronic Control Concepts, 2002.

[7] J. W. Jeon, and Y. G. Kim, ”FPGA Based Acceleration and Deceleration Circuit for Industrial Robots and CNC Machine Tools,” *Mechatronics*, Vol. 12, Issue 4, pp. 635-642, May. 2002.

[8] J. W. Jeon, ”Method for Controlling the Traveling Path of a Robot during Acceleration and Deceleration,” United States patent, Patent Number: 5,373,439, Dec. 13, 1994.

[9] J. W. Jeon, ”An Efficient Acceleration for Fast Motion of Industrial Robots,” *Proceedings of 1995 IEEE 21st IECON,* Orlando, FL, pp. 1336-1341, Nov. 6-10, 1995.