System Design Consideration for Digital Wheelchair Controller

Ruei-Xi Chen, Liang-Gee Chen, *Senior Member, IEEE*, and Lilin Chen

***Abstract*—The design and implementation of a digital wheelchair controller system is presented in this paper. This novel model depicts an information flow between the driving commands and wheel speed. A command interpreter and two speed processing datapaths are proposed as functionally independent blocks for the controller. The control process consists of the following steps: command decoding, speed estimation, and speed serving. Through proper partitioning to concurrent blocks, the design complexity is reduced significantly. Block reconfiguration for field-programmable gate array rapid prototyping is also employed in this paper, and system fidelity and development efficiency are revealed through the experimental results.**

***Index Terms*—Block design, electrical differential gears, field-programmable gate array, proportional integral derivative control, pulsewidth modulation, rapid prototyping, rotary encoders, speed estimators, speed models, voice driven, wheelchair controllers.**

I. Introduction

THE electric controller is the major signal-processing kernel in a wheelchair system. It contains transfer functions for converting driving commands to motor power. The goal of the design is aimed at regulating the wheelchair speed for user comfort. Several approaches, based on analog technologies, to this problem have been taken. Traditionally, analog technologies composed of signal processing and interfacing are commonly employed. Recently, digital systems based on microprocessor controlling, have become attractive research topics. An adaptable optimal control for dc motors in a wheelchair using a microprocessor system is proposed in [1]; an intelligent control-model based on a micro-controller is presented in [2]; an advanced system with autonomous mobility based on PC-486 notebook computer is described in [3]; and a study for programming the voice-controlled wheelchair system is illustrated in [4]. Among these approaches, the program-based system gains the benefit of being able to easily upgrade the control strategies. However, it causes a new problem. The real-time performance must be taken into consideration while implementing the concurrent functions with time sharing methodologies.

In this paper, a novel model to address the design problem of a digital wheelchair controller is proposed. This model, which specifies the information flow between driving command and wheel speed, is composed of three functions: the command interpretation, the speed estimation, and the speed servo blocks. Block architectures corresponding to the above functions are demonstrated. This approach is considered suitable for field-programmable gate array (FPGA) rapid prototyping [5], [6]. The function blocks, configured by the FPGA, can be treated as the concurrent and programmable hardware modules, gaining both efficiency and flexibility in system design.

This paper is organized as follows. A novel model for a control strategy is described in the next section. The block-based system specification and the proposed block architectures are then presented in Section III. Furthermore, the design approach using FPGA prototyping and the simulation results are illustrated in Section IV. Finally, the realization is presented and the conclusions are given in Section V.

Manuscript received December 4, 1998; revised September 23, 1999. Abstract published on the Internet April 21, 2000. This work was supported by the Chip Implementation Center, Taiwan, R.O.C., under Project NSC87-2215-E-002-028. This paper was presented at the 10th VLSI Design/CAD Symposium, Nan-Tou, Taiwan, R.O.C., August 18–21, 1999.

The authors are with the DSPIC Laboratory, Department of Electrical Engineering, National Taiwan University, Taipei, Taiwan, R.O.C.

Publisher Item Identifier S 0278-0046(00)06813-1.

II. System Models

The major function of an electric wheelchair controller is the processing of the driving command and the adapting of motor power. Such a system contains multilevel information processing schemes and can be referred to as one type of a motion mechatronics (mechanics and electronics) system [7]. Variables of this system, including position, speed, torque; and driving commands flow among the user, sensors, and actuators. Above all, the main control variable is the speed of the wheels [2], [8]. Loading, friction, and road conditions, such as rough surfaces and slopes, are factors influencing speed [1]. Control algorithms, employed to smooth speed change, will be accomplished in the following model.

A. Control Model

Most wheelchair mechanisms only power the rear wheels. Therefore, while neglecting the friction effect, the speeds of the rear wheels becomes the major controlling variables. In this design, the speed of both wheels is controlled by the driving commands from acquisition facilities. Such a command-driven system is simply modeled as in Fig. 1(a). The movement of the wheelchair can be represented as the vector sum of the rear wheels' speeds and can be expressed as the speed model shown in Fig. 1(b). In this model, the horizontal and the vertical axes represent the speeds of the left wheel and the right wheel, respectively. Here, a positive value means forward movement and a negative value represents backward movement. The combinations of both wheel speeds from quadrant one to four indicates forward motion, left circular motion, backward motion, and right circular motion, sequentially. The speed combination

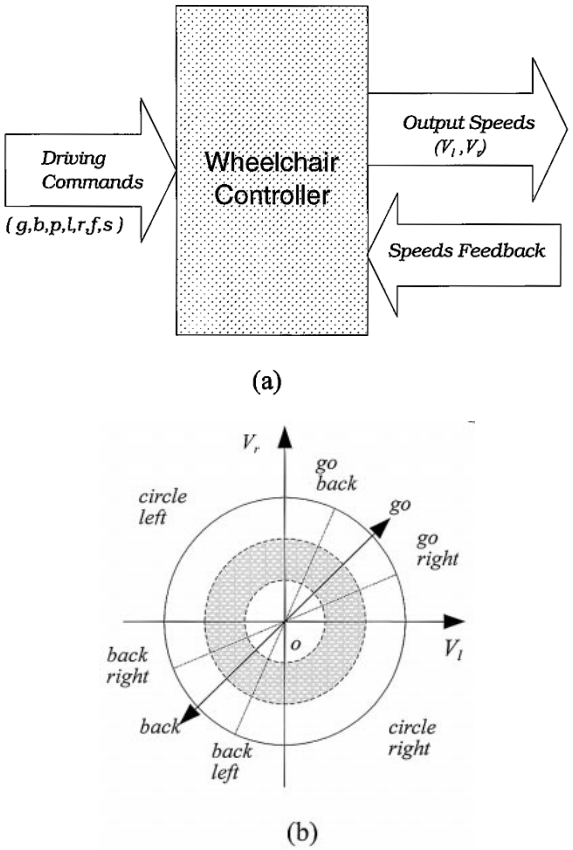

Fig. 1. Models of wheelchair controller. (a) Control model. (b) Speed model.

at 45° indicates direct forward motion and that at the 225° represents direct backward motion. The chair will turn left or right if the combination of both wheel speeds falls in the other parts of quadrant one and three.

In analog approaches, the speed model is usually mapped directly to the joystick's behavior and controlled by continuous voltage. In a digital system, the speed and the direction corresponding to discrete commands are determined according to the cross point of lines and circles as in Fig. 1(b).

The control process of the wheelchair can then be divided into three steps: command interpreting, speed estimation, and speed serving.

B. Command Interpreter

Driving commands are usually issued by the acquisition facilities of the wheelchair such as joysticks, head movement sensors [9], or a voice recognizer [4], [10], [11]. Most of the driving-commands fall into two categories. The *jump-type command* will load a fixed speed using *stop(p)*, *go(g)*, and *back(b)* commands. The *step-type command* will change current speed and direction step by step according to *left(l)*, *right(r)*, *faster(f)*, and *slower(s)* commands. Here, the command interpreter converts the output of the acquisition facility to appropriate control signals for the following datapaths.

C. Speed Estimator

The speed estimator is the first stage in speed-processing datapaths, which sets up the desired speed according to the driving command and the current estimated speed. The desired speed v_{x_esti} can be obtained through the summation of the *initial speed* (loaded value) and *the accumulation of n-step variations* δ_x. Equation (1) illustrates this

$$v_{x\_\mathrm{esti}}(n) = \delta_x(0) + \sum_{i=1}^{n} \delta_x(i). \tag{1}$$

Here, $v_{x\_\mathrm{esti}}(n)$ is the speed at the nth sampling time, $\delta_x(0)$ is the initial speed given by the *jump-type command*, and the subscript x represents left (l) or right(r). The $\delta_x(i)$ represents *the ith speed variation* corresponding to the *step-type command* as shown in Table I. Here, the constant K indicates the initial speed value and the constant d is the step value with a minimum value one. In Table I, nc means no command is detected at sampling time, therefore, no speed change is required and can, thus, be categorized as *the step type command*. Furthermore, index n must be reset whenever the *jump-type command* is issued.

Equation (1) can be normalized by a three-valued function $f_x(c)$, $+1$, 0, and -1, according to Table II and can be rewritten as the following equation:

$$v_{x\_\mathrm{esti}}(n) = K \cdot f_x(c_{\mathrm{jump}}(0)) + \sum_{i=1}^{n} d \cdot f_x(c_{step}(i)). \tag{2}$$

This is the general case for wheel speed notation. In a joystick system, the speed model can be simplified to the second term since no *jump-type command* will be issued. However, the limitation of the accumulator must be taken into account. From the view point of implementation, (2) is expressed in the recursive form as

$$v_{x\_\mathrm{esti}}(n) = \begin{cases} K \cdot f_x(c_{\mathrm{jump}}(0)), & \text{for jump command} \\ v_{x\_\mathrm{esti}}(n-1) & \\ \quad +d \cdot f_x(c_{step}(n)), & \text{if step command and} \\ & |v_{x\_\mathrm{esti}}(n-1)| \\ & \le (\mathrm{Max.} - d) \\ v_{x\_\mathrm{esti}}(n-1), & \text{if step command and} \\ & |v_{x\_\mathrm{esti}}(n-1)| \\ & > (\mathrm{Max.} - d). \end{cases} \tag{3}$$

Here, Max is the positive speed limit. In (3), whenever the *jump-type command* is issued, the estimated speed $v_{x\_\mathrm{esti}}(n)$ will be set to plus/minus K or zero and the index n to zero. The second case represents a normal speed change corresponding to the step command, while the third case is overflow caused by the speed limit, and the estimated speed $v_{x\_\mathrm{esti}}(n)$ will remain the same.

The speed estimator can be viewed as an electrical differential gear, which distributes the speeds of separate wheels. For example, when the "right" command is issued, the speeds will be updated based on (3)

$$\underline{v(n)} = \begin{bmatrix} v_l(n) \\ v_r(n) \end{bmatrix} = \begin{bmatrix} v_l(n-1) \\ v_r(n-1) \end{bmatrix} + d \begin{bmatrix} 1 \\ -1 \end{bmatrix}. \tag{4}$$

From the above equation, the left and the right speed will be increased and decreased by d, respectively. Eventually, the left wheel will turn faster than the right wheel, causing the wheelchair to turn right.

TABLE I
RELATION OF COMMANDS AND DESIRED WHEEL SPEED CHANGE VALUES

Command Types / Change Value	Jump type commands (Load constant speed value)			Step type commands (Change speed by one step value)				
	c_0,	c_1	c_2	c_3	c_4	c_5	c_6	c_7
	stop	*go*	*back*	*left*	*right*	*faster*	*slower*	*nc*
δ_l	0	K	-K	-d	d	d	-d	0
δ_r	0	K	-K	d	-d	d	-d	0

TABLE II
THREE-VALUE COMMAND TRANSFER FUNCTION

$f_x(c_i(n))$ / $c_i(n)$	$C_{jump}(n)$			$C_{step}(n)$				
	c_0,	c_1	c_2	c_3	c_4	c_5	c_6	c_7
$f_l(c_i(n))$	0	1	-1	-1	1	1	-1	0
$f_r(c_i(n))$	0	1	-1	1	-1	1	-1	0

D. Speed Servo

Because a rapid speed change of the wheelchair might result in discomfort or even danger to the user, adaptive functions for a smooth speed change must be employed. An accelerator and a speed adapter are designed to accomplish these.

1) Accelerator: Once a new speed $v_{x\_esti}(n)$ is given, the set speed $v_{x\_set}(m)$ will be accelerated to the estimated value. The behavior of accelerating functions can be expressed as

$$
\begin{aligned}
&if\,(v_{x\_esti}(n) > v_{x\_set}(m))\,then\{\\
&\quad v_{x\_set}(m+1) = v_{x\_set}(m) + a; \quad //\text{accelerating.}\\
&\}\\
&else\,if(v_{x\_esti}(n) < v_{x\_set}(m))\,then\{\\
&v_{x\_set}(m+1) = v_{x\_set}(m) - a; \quad //\text{decelerating.}\\
&\quad \}\\
&else\{\\
&\quad v_{x\_set}(m+1) = v_{x\_set}(m); \quad //\text{no change}\\
&\}
\end{aligned}
$$

where a is the accelerating step, m and n is a set resulting from the different sampling rates in the speed accelerator and speed estimator.

2) Speed Adapter: The speed adapter is used to adapt the output power. There are several design strategies to be considered.

a) If this function is eliminated, it becomes an open-loop speed control system.

b) The use of proportional integral derivative (PID) control theory is to be considered.

PID control is widely used in the industry. The PID control algorithm based on [12] can be rewritten in the following discrete and recursive form:

$$
\begin{aligned}
&m(n) = K_p \cdot e(n) + K_i \cdot S(n) + K_d \cdot d(n)\\
&\text{where} \quad S(n) = S(n-1) + e(n)\\
&\text{and } d(n) = e(n) - e(n-1). \qquad (5)
\end{aligned}
$$

Here, $m(n)$ denotes the controlling power. The error function $e(n)$ is obtained from the difference between the set speed $v_{x\_set}(m)$ and the feedback speed $v_{x\_f}(m)$. The three control parameters (K_p, K_i, and K_d) must be carefully tuned because they affect the system's reaction and stability.

c) Other algorithms can be considered.

The performance of the speed adapter can be further enhanced through other advanced control methods. In self-tuning PID control, the parameters can be automatically adjusted to achieve the best response [13], [14]. Other algorithms, such as a phase-locked loop (PLL) motor speed control [15], can also be employed for advanced applications.

III. SYSTEM SPECIFICATION AND PROPOSED ARCHITECTURE

A. General Description of Electric Wheelchair System

An electric wheelchair can be partitioned into the function blocks shown in Fig. 2. The major blocks contain acquisition, control, and power. The acquisition block, composed of the joystick, the head movement sensor, or the voice recognizer, extracts the manipulator's commands. The control block, which converts the driving commands to motor power, includes three subblocks: the command interpreter, the speed estimator, and the speed servo. The power block is composed of the H-bridge amplifiers, which power pulsewidth modulation (PWM) signals to the dc motors.

Interface buses labeled "A"–"F" act as block connectors. Generally speaking, buses are not necessarily carrying only the electric signals. The paths "A" and "F" are the physical information flow associated with the user's reaction such as commands from the hand, head, mouth, or eyes. Bus "D" contains the energized PWM signals and bus "E" carries the feedback information detected by the speed sensors. Through the interface of these buses, the system forms a closed-loop control flow.

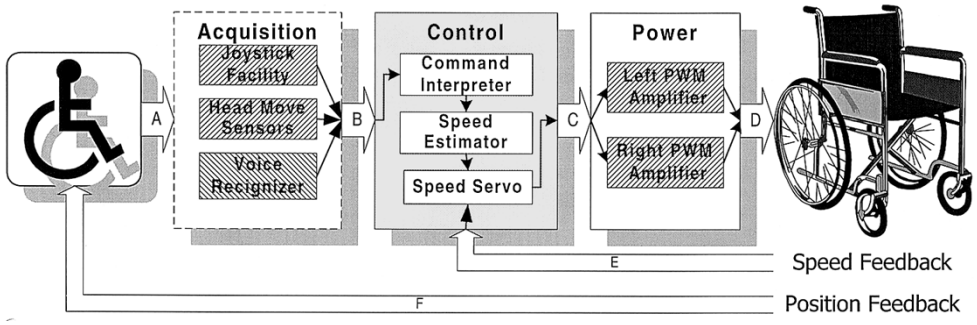

Fig. 2. Block diagram of the wheelchair controller system.

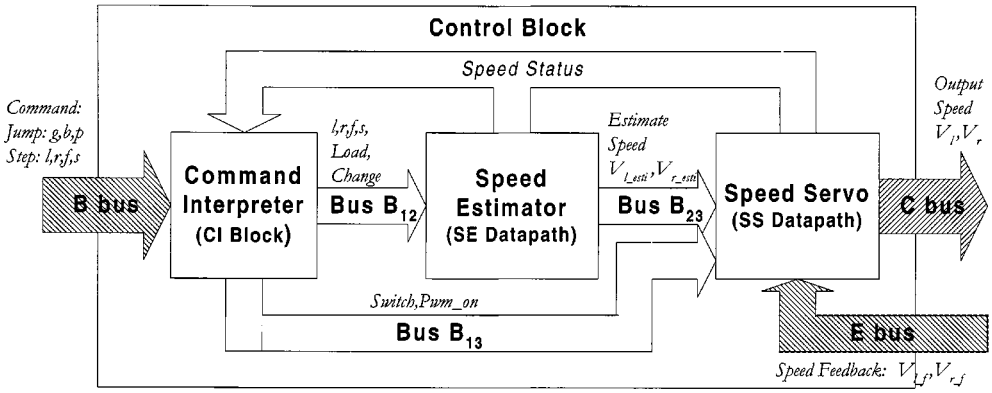

Fig. 3. Architecture of the control block.

B. Proposed Architecture for the Control Block

The control block, as seen in Fig. 3, is constructed mainly with the connection buses and the three processing elements. The command interpreter converts the commands to proper control signals: the speed change information for bus "B_{12}" or the motion conditions for bus "B_{13}." The speed estimator generates the estimated speed and sends it to bus "B_{23}." The speed servo corrects the error between the estimated speed in bus "B_{23}" and the feedback speed in bus "E" and then generates adequate PWM signals to "C" bus. The command interpreter is composed of command registers and a finite-state machine (FSM) for decoding. The function of this unit is similar to that of the control unit in a microprocessor.

The datapaths of the speed estimator (SE) and speed servo (SS) are shown in Fig. 4. The left and right SE datapaths are symmetrical in order to control the electrical differential gear, as shown in Fig. 4(a). The adder and the accumulator are major speed processing components. The lookup table and the multiplexer determine the control function. The function of (3) will be implemented through this architecture.

Fig. 4(b) shows the one-channel architecture of the SS datapaths, which is composed of the accelerator, PID controller, and PWM converter. The accelerator will work as follows. If the set speed is not equal to the estimated speed, the current value will be increased or decreased by one through the adder. Otherwise, the set speed will remain the same. The PID control block introduces a parallel architecture for three-term computing by the accumulator, delay element, adders and multipliers. The pipelined structure is considered to maximize the clock rate. Note that the data width will be doubled after multiplication. The PWM converter is implemented using a counter and a digital comparator. The result is converted to H-bridge PWM control signals for energizing the dc motors.

In an open-loop control, the set speed can be connected directly to the PWM converter without a PID block. The different combinations of P, PI, and PD controls can also be considered. The speed feedback signals in the "E" bus must be exploited while the closed-loop control is implemented.

IV. Experimental Results and Discussion

A. FPGA Simulation

The 10K20 FPGA, the device suitable for prototyping the aforementioned wheelchair controller, consists of 1152 Logic Cells (LCs), equivalent to 15 k$\sim$63 k usable gates depending on applications [16]. This device provides 144 logic array blocks (LABs) and six embedded array blocks (EABs). Each LAB block therefore contains 8 LCs. Block design and validation can be easily realized in hardware description language (HDL) and integrated into a graphic environment. The results after logic synthesizing, technology mapping, and device fitting are reported in Table III. The utilization of LAB reveals the complexity of each block. PID control is the largest area block, occupying approximately 73% of the system. The data width in the SE/SS datapaths is 8 bits and that in the PWM converter is 8/16 bits.

Design fidelity is the most important consideration in this design. A simplified circuit of 4-bits width with an open-loop control is tested for design fidelity, and the results are shown in

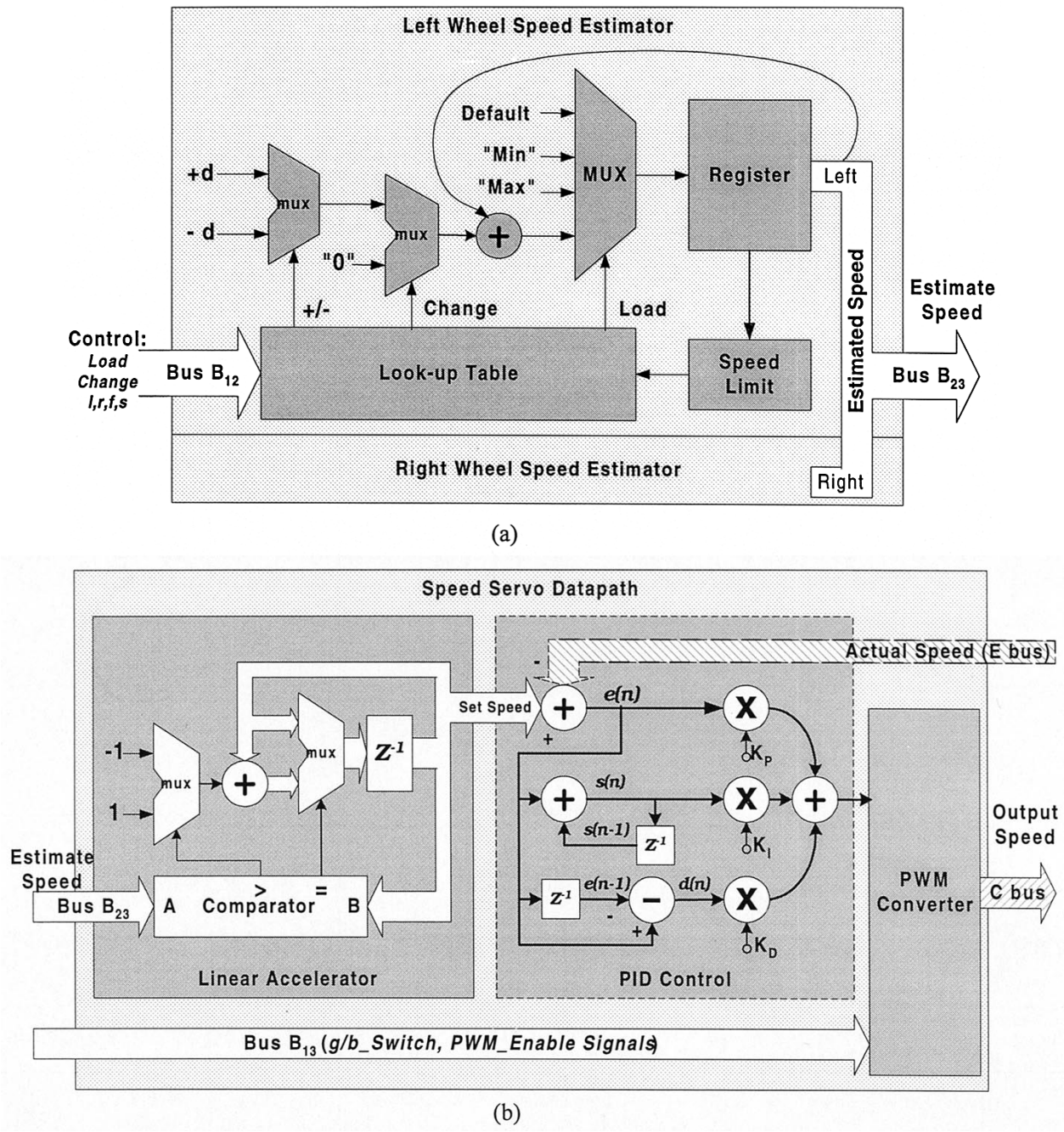

Fig. 4. Architecture of speed estimator and speed servo blocks. (a) Speed estimate datapath. (b) One channel of left/right wheel speed servo datapath.

TABLE III
AREA UTILIZATION FOR FPGA IMPLEMENTATION

	LCs	Percent for Blocks	% Utilization in 10k20
Timing Generator	47	4.7%	4.1%
Command Interpreter	42	4.1%	3.6%
Speed Estimator	71	7.0%	6.2%
Speed Servo_Accerator	76	7.5%	6.6%
Speed Servo_PID	740	73.3%	64.2%
PWMs	34	3.4%	3%
Total	1010	100%	87.7%

Fig. 5. At first, the "simu" signal is used to shrink the time scale from the actual time. Then, the seven driving commands are issued sequentially to test their functions. The following abbreviations of the states in the command interpreter FSM include the initial state (*init*), the power switching state (*sw*), the loading state (*lds*), the running state (*run*), and the speed-changing state (*chg*). Here, the *lds* state is caused by the *jump-type command*, and the *chg* state corresponds to the *step-type command*. In this test, the constant K and the step value d are set to 4 and 2, respectively.

In Fig. 5, the test commands are issued as the sequences of *go (start)*, *left*, *left*, *go (restart)*, *faster*, *faster*, *slower*, *right*, and

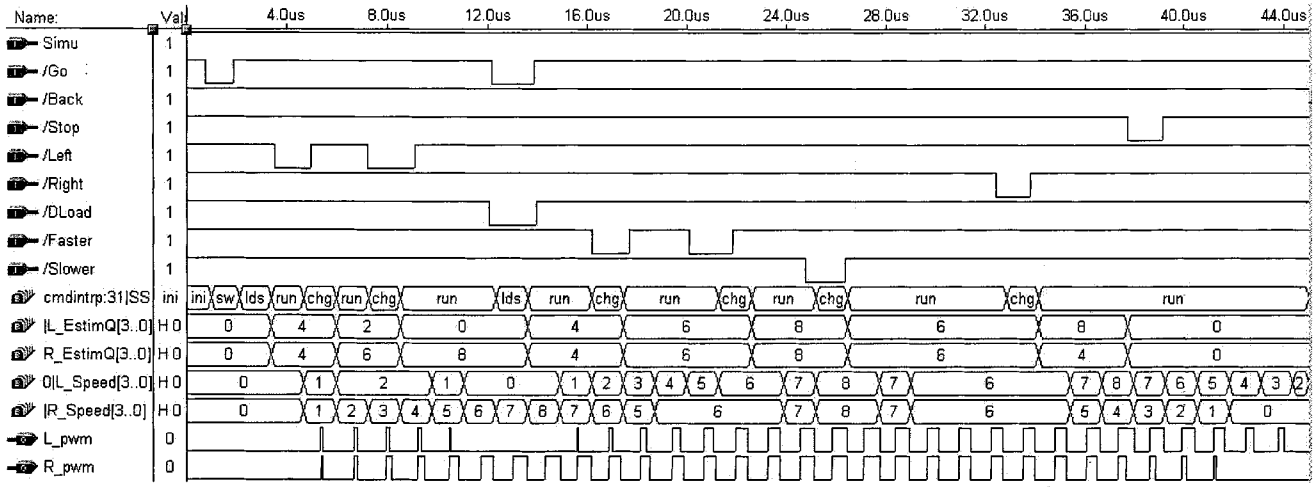

Fig. 5. A brief section of PWM waveforms corresponding to driving commands (note that the time scale is shrunk).

stop, etc. The estimator will load the left and the right estimation registers with a constant K ("4") while the *go command* is issued. Then, the speed servo accelerates the output speed as well as the pulsewidth of PWM signals. When *the left command* is issued, the left and the right estimated speeds are set to 2 and 6, respectively, while the second *left command* is issued, the left estimated speed will be reduced to zero and the right estimated speed increased to 8. Note that the PWM signals will soon respond to this change through an accelerated process. The *faster command* increases both wheel speeds by 2. The *slower command* acts in reverse to the *faster one*, and so does the *right command* to the *left one*. The *stop command* loads both of the estimation registers zero value and causes the wheelchair to decelerate to a complete stop.

B. FPGA Implementation

The FPGA controller is realized in two versions. Version one is a simplified 4-bits-width open-loop controller implemented in a CPLD 7064 chip. The other version includes the PID scheme utilized in our FLEX-10k40 ISA-interface card as shown in Fig. 6(a). This card is developed to ease PC interfacing during the design and the test phase, and the FPGA controller to work independently upon completion of the entire function test. A configuration ROM, up to 2 Mbits, can be used for downloading the control circuit to the FPGA device as power on. An on-board 25.175 MHz oscillator is used in the FPGA card. A 40-pin I/O port is used to interface the controller to acquisition and power blocks. Within the FPGA, a command interpreter, speed estimators, accelerators, PID controllers, actual speed measurers, data representation converters, and PWM converters are accommodated. The data before the PWM converter is a 2's complement representation, which afterwards is converted to a sign-magnitude representation for easing the generation of the full H-bridge PWM control signals. A clock generator is designed to provide the state clock (128 Hz), the acceleration clock (32 Hz), and the sampling clock for actual speed measurement. The control cycle is set at 1/128 s.

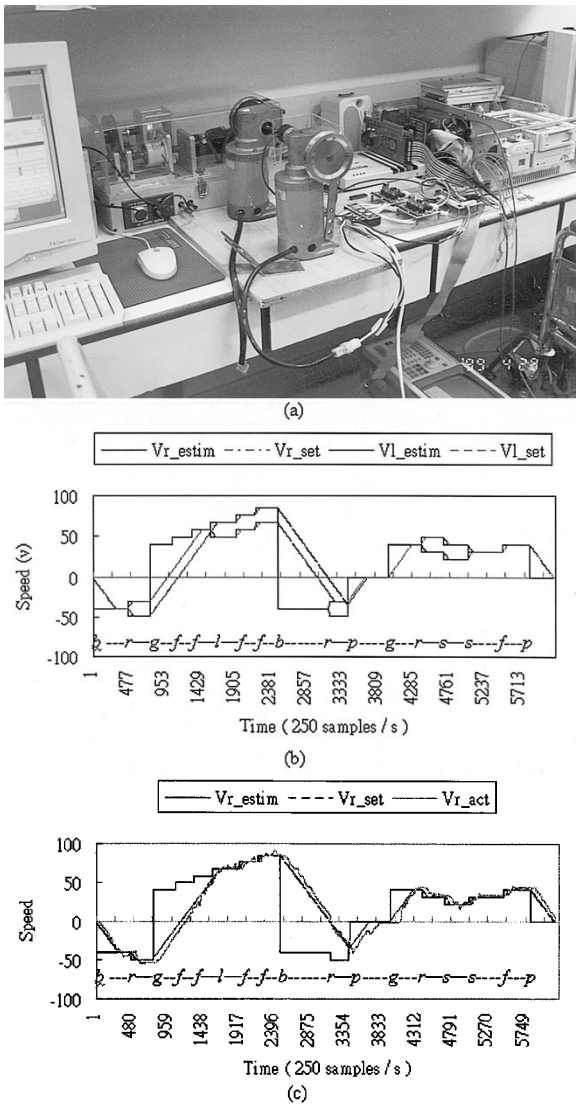

Fig. 6. FPGA wheelchair controller. (a) Development environment. (b) Estimated and set speeds for right and left wheels. (c) The actual speed response of right wheel. (Time scale = 1 s.)

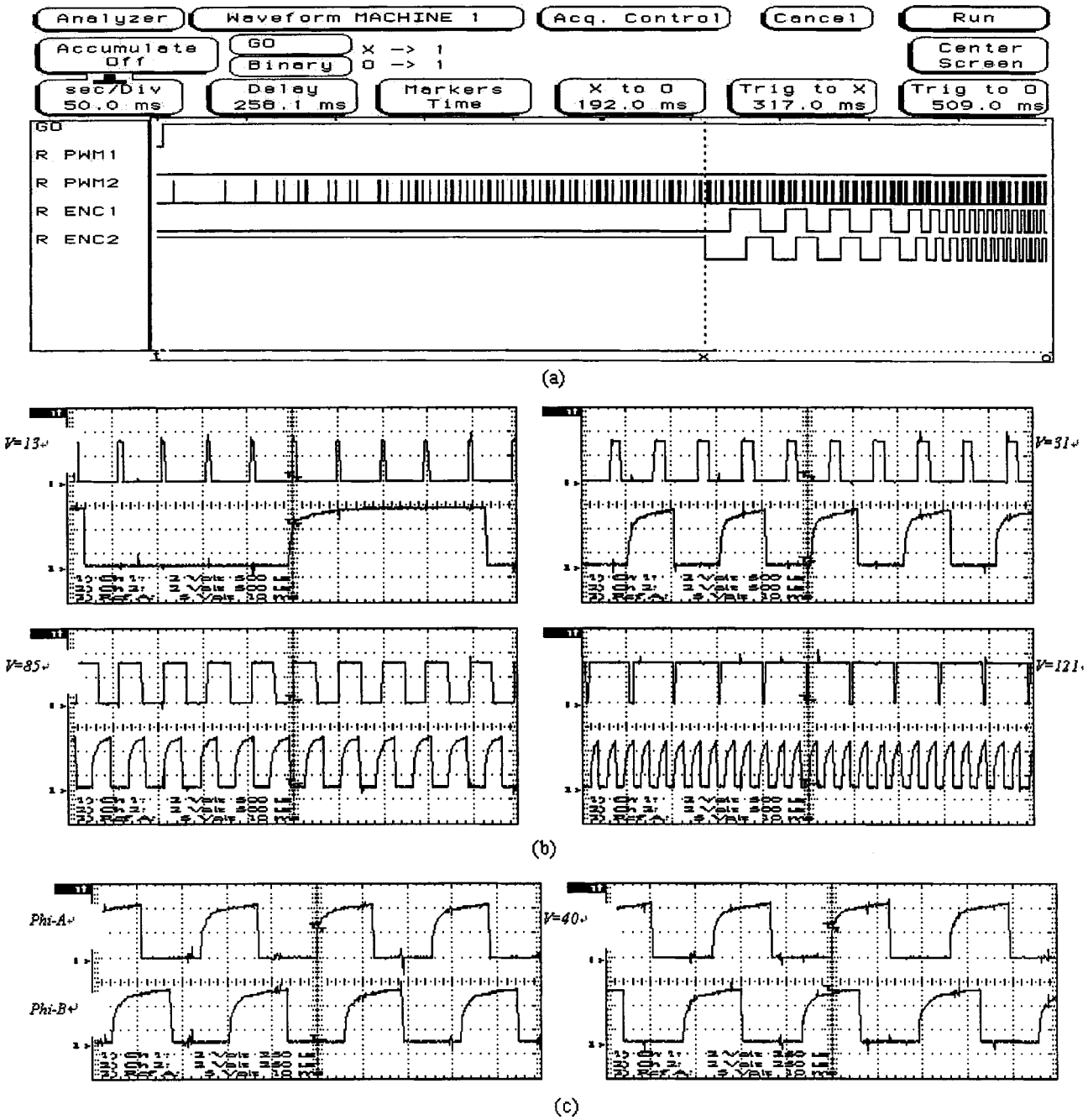

Fig. 7. Waveforms of PWM and filtered rotary encoder signals. (a) Responses after the go command issued. (b) Encoder speed following the PWM pulsewidth (2 V 500 μs/div). (c) Forward and backward encoder signals (2 V 250 μs/div).

In this design, the data width is set as 8 bits, thereby the active range of speed is from -128 to $+127$. The load value K corresponding to the *jump-type command* is set at 40, and the step value d corresponding to the *step-type command* is set at 9. The test results are shown in Fig. 6(b) and (c). The speeds are obtained from the PCs I/O using an interface control program. Such an environment is shown in Fig. 6(a). The measuring cycle is 4 ms and 6164 data sets are measured in the total time of 25 s. Fig. 6(b) reveals the fidelity of the FPGA controller. Following a series of driving commands, listed in the figure, the estimated and accelerated wheel speeds from the FPGA are measured. In Fig. 6(c), only the right wheel speed is presented, however, the actual speed response is included. The figure depicts the actual speed, which does not follow the estimated speed V_estim but rather the accelerated speed V_set.

The waveform of PWM signal and the response of the rotary encoder corresponding to the driving command "go" are shown in Fig. 7(a). In this figure, the pulsewidth of the PWM signal is increasing until it reaches the set value "K." Here, the frequency of the PWM signal is 2.048 kHz. The nonuniform waveform results from the analyzer's sampling loss to a small width of the PWM pulse. The motor starts running after 317 ms delay. At that time, the rotary encoder activating and the frequency of its output signals is proportional to the pulse width of PWM signal,

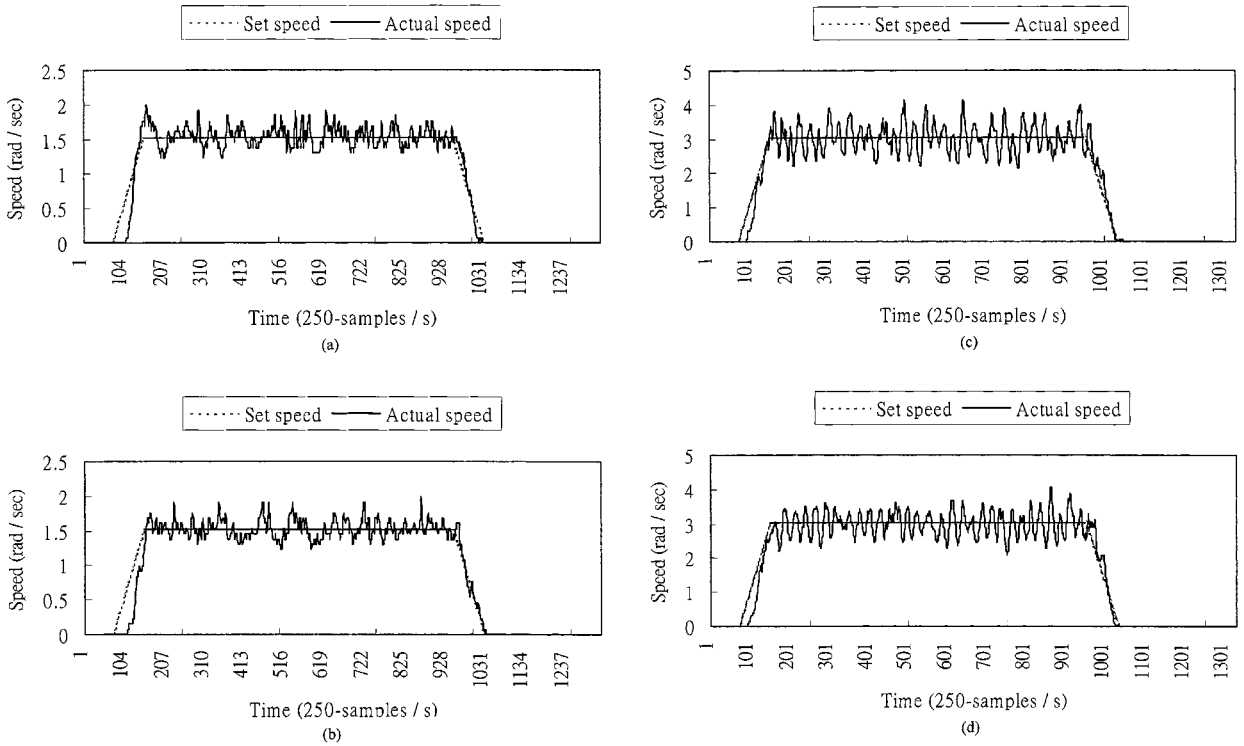

Fig. 8. Right motor test for different speeds: 1.5 rad/s of (a), (b) versus 3.0 rad/s of (c), (d), and different weights: 25 kg of (a), (c) versus 75 kg of (b), (d).

so as to the set speed. The relation is shown in Fig. 7(b), while, in Fig. 7(c), the direction can be determined from the leading edge of the two output signals of the rotary encoder.

The actual speed responses are measured under various loads and speeds. The experimental results reveal that the performance fits to the design goal independently of the rider's weight and the driving speed. In Fig. 8, the actual speeds of the right motor under different riders (25/75-kg weights) and set speeds (1.5/3.0 rad/s) are displayed. Comparing Fig. 8(a) to Fig. 8(b) and Fig. 8(c) to Fig. 8(d), the lighter of the riders will result in the faster response, but all speed changes are within 30% range. The beginning of each new command, which usually causes the rapid change of speed and results in discomfort, has been carefully dealt with. The estimated speed for each individual command is converted to the set speed by the speed accelerator. The major consideration for the set speed is to smooth the change of wheels running according to the characteristics of the wheelchairs. Through this mechanism, the actual speeds can be smoothly adjusted for all cases, regardless of loads and speed settings.

In Fig. 9(a) and (b), which expands the starting and stopping parts of Fig. 8(c), the output power values to the PWM converter are further evaluated. It shows that the adapting of the controlling power is not affected by the speed changes. Thus, the smooth adjustment of wheel running according to the new commands can be assured.

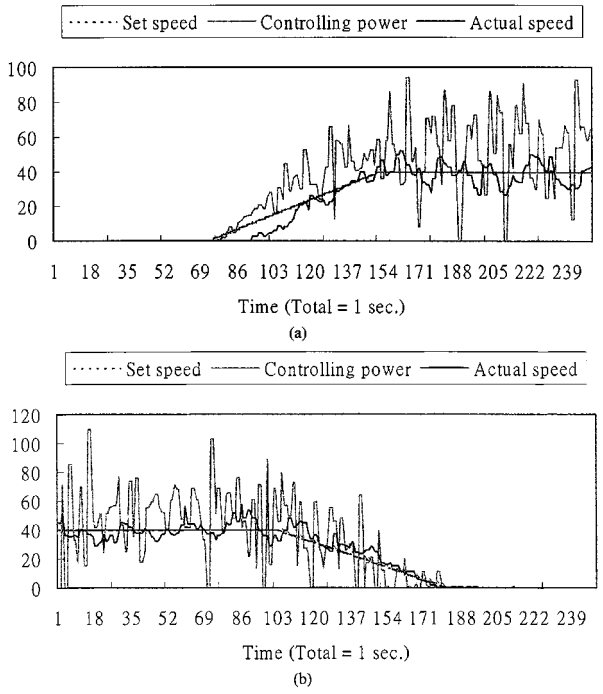

Fig. 9. Relation of controlling power and speeds. (a) 1 s of the start. (b) 1 s of the end. (Controlling power and the speeds are represented by 8-bit signed digital values. Here, the speed value 40 specifies 3.0 rad/s in this design.)

C. Discussion

Certain problems make the controller's implementation difficult. First, the rotary encoder is normally used to detect the actual speed that has two directions. Two square waves, A and B, are output with one having a 90° phase lead to the other. The rotational direction is determined according to what is the leading one. However, these signals are noisy and high-frequency noise must be filtered out. Otherwise, a speed-calculation and direction-detecting error will result. It will impede the PID block from generating the proper power to energize the motor, and

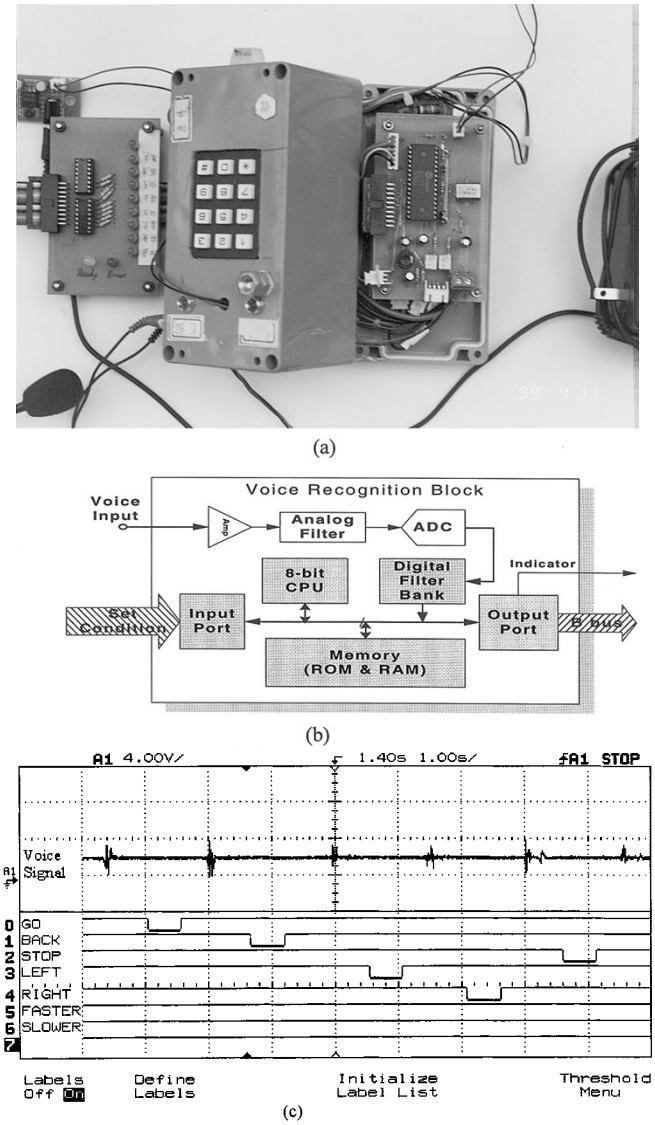

Fig. 10. Response of voice recognizer. (a) Photograph of the PC board. (b) Block diagram of the voice recognizer. (c) Test result. (The tested voice signal is issued by the author with the sequence of "go, back, left, right, stop," etc.)

the H-bridge driver then may be destroyed. Second, the motors must be carefully chosen to fit the mechanical requirement. To drive the wheelchair, it is often necessary to increase the torque of the motor through one or two stages of speed decreasing gears. The stage that the rotary encoder is locating will determine the sampling rate. For example, the maximum speed of the dc motors used in this design is 4500 r/min and a two-stage speed-decreasing gear with 12 : 1 and 4 : 1 ratios is equipped. The diameter of the wheels used is 50 cm. Thus, the maximum speed of the wheelchair will be approximately 2.5 m/s. Assume that the 1-k pulses/revolution type of rotary encoder is mounted in the first stage of the speed decreasing gear, a 6.25-k pulses per second (PPS) signal will eventually be sent out. However, if the rotary encoder is connected to the motor directly, then 75-k PPS will be sent out. The counting circuit and the sampling clock must be set according to this specification to prevent overflow. Another limitation is the lowest speed detection value corresponding to the minimum speed, which must be taken into account to prevent a zero zone from occurring. A circuit for doubling or quadrupling the output frequency of the rotary encoder is necessary for shrinking the zero zone and increasing the resolution.

Fig. 11. Disabled person, K. B. Su, tests the fidelity of the controller.

The third problem is the issuing rates of the driving commands. In the design, although the processing speed of FPGA circuit may up to 30 MHz, the control cycle is set to 128 Hz. Thus, the rates of the driving commands need not exceed this limit. Fortunately, for voice-driven equipment, the command generation rate is very slow. Fig. 10 shows the strobe response of a speaker-dependent voice recognizer [11] employed in the design. The recognition process required about 0.5 s, although it is reported as 33 ms. However, the real delay is caused by human reaction, in that the voice commands are issued too late for turning around an obstacle, which will require more than 0.5 s for disabled people, causing the driving to be not as smooth as it could be. Fig. 11 shows the driving test of a disabled person, K. B. Su, who indicated that the intelligent controller, meaning the automatic sensors and drives schemes, should be taken into account in the design to allow more comfortable driving.

According to the aforementioned approach, the wheelchair controller is partitioned into three independent functional blocks: command interpreter, speed estimator, and speed serving processor. This makes the function enhancement of any individual block more easily be adopted and verified. In a digital integrated system, an FPGA implementation can be viewed as a subset of an ASIC design. A block-based approach for rapid prototyping can ease the system migration. The most meaningful issue for this presentation is the establishment

of parameterized reuse libraries. The parameterized modules enable efficient mapping, which converts digital designs into divergent architectures implemented using various technologies. In this system, we construct the datapaths with a scalable parameter referred to as "width." Therefore, the word lengths are easily reconfigured to different bits (such as 4, 8, 16, etc.) by simply changing this parameter.

V. Conclusion

A novel approach to a real-time digital system design for an electric wheelchair controller has been described and developed. The design is simplified by partitioning the system into several independent functional blocks. A command interpreter and two signal processing datapaths are concurrently functioning to fulfill the controller. The control process is constructed through the following steps: command decoding, speed estimation, and speed serving. The block based design approach maintains the advantages of reducing design complexity and simplifying validation. The characteristics of the FPGA on the flexibility of hardware reconfiguration and the utilization of parameterized library resources make it suitable for the block-based design. The success of the FPGA rapid prototyping in its application to this wheelchair controller design has revealed its fidelity and efficiency.

Acknowledgment

The authors would like to thank K. B. Su, who helped to complete the test work and suggested some meaningful discussions. The prototype of the voice-driven wheelchair controller based on the proposed approach was approved by the Taiwan Eden Social Welfare Foundation in July 1998.

References

[1] K. E. Brown, R. M. Inigo, and B. W. Johnson, "Design, implementation, and testing of an adaptable optimal controller for an electric wheelchair," *IEEE Trans. Ind. Applicat.*, vol. 26, pp. 1144–1157, Nov./Dec. 1990.

[2] R. A. Cooper, "Intelligent control of power wheelchairs," *IEEE Eng. Med. Biol. Mag.*, vol. 14, pp. 423–431, July 1995.

[3] G. Bourhis and P. Pino, "Mobile robotics and mobility assistance for people with motor implements: Rational justification for the VAHM project," *IEEE Trans. Rehab. Eng.*, vol. 4, pp. 7–12, Mar. 1996.

[4] K. B. Stanton *et al.*, "PSUBOT—A voiced-controlled wheelchair for the handicapped," in *Proc. 33rd Midwest Symp. Circuits and Systems*, vol. 2, Calgary, AB, Canada, 1991, pp. 669–672.

[5] S. Brown and J. Rose, "FPGA and CPLD architectures: A tutorial," *IEEE Des. Test Comput.*, vol. 13, pp. 42–56, Summer 1996.

[6] M. Donlin,, "CPLD/FPGA devices, tools lure PLD designers into faster, denser logic," *Comput. Des.*, pp. 81–97, Nov. 1995.

[7] R. Isermann, "On the design and control of mechatronic systems—A survey," *IEEE Trans. Ind. Electron.*, vol. 43, pp. 4–15, Feb. 1996.

[8] R. A Cooper, *Rehablitation Engineering Applied to Mobility and Manipulation*. Philadelphia, PA: Adam Hilger IOP, 1995.

[9] D. Jaffe, H. L. Harris, and S. K. Leung, "Ultrasonic head controlled wheelchair interface: A case study *in* development and technology transfer," in *Proc. 13th Annu. RESNA Conf.*, Washington, DC, 1990, pp. 23–24.

[10] G. Miller, "Voice recognition as an alternative computer mouse for the disabled," in *Proc. 15th Annu. RESNA Conf.*, Toronto, ON, Canada, 1992, pp. 58–59.

[11] *Commercial ICs—Miscellaneous ICs*, UMC Inc., Hsin-Chu, Taiwan, R.O.C., 1997, pp. 5-3–5-51.

[12] S. Bennett, *Real-Time Computer Control: An Introduction*. Englewood Cliffs, NJ: Prentice-Hall, 1988.

[13] B. Wittenmark and K. J. Astrom, "Simple self-tuning controller," in *Methods and Applications in Adaptive Control*, H. Unbehanen, Ed. New York: Springer-Verlag, 1980.

[14] R. Ortega and R. Kelly, "PID self-tuners: Some theoretical and practical apsects," *IEEE Trans. Ind. Electron.*, vol. IE-31, pp. 332–338, Nov. 1984.

[15] G. C. Hsieh and J. C. Hung, "Phase-locked loop techniques—A survey," *IEEE Trans. Ind. Electron.*, vol. 43, pp. 609–615, Dec. 1996.

[16] *Altera Data Book*, Altera Inc., San Jose, CA, 1998.

Ruei-Xi Chen was born in Nan-Tou, Taiwan, R.O.C., in 1954. He received the B.S. degree from the Electronic Engineering Department, Tamkang College, Taipei, Taiwan, R.O.C., the M.S. degree from the Information Engineering Department, Tamkang University, Taipei, Taiwan, R.O.C., and the Ph.D. degree form the Electrical Engineering Department, National Taiwan University, Taipei, Taiwan, R.O.C., in 1975, 1982, and 2000, respectively.

He has been a Lecturer with St. John and St. Mary Institute of Technology, Taipei, Taiwan, R.O.C., since 1982. His students have received 12 awards in the Taiwan National Competition of Microcomputer Hardware Design. He was a Q.A. Engineer with Sun-Moon-Star Communications Company, Taipei, Taiwan, R.O.C., from 1979 to 1980. He has assisted several companies in establishing key technologies in microprocessor applications. His research interests include speech coding, digital signal processing, media processors, VLSI architecture design, reconfigurable computing, and system implementation methodologies.

Liang-Gee Chen (S'84–M'86–SM'94) was born in Yun-Lin, Taiwan, R.O.C., in 1956. He received the B.S., M.S., and Ph.D degrees in electrical engineering from National Cheng Kung University, Tainan, Taiwan, R.O.C., in 1979, 1981, and 1986, respectively.

He was an Instructor (1981–1986), and an Associate Professor (1986–1988) in the Department of Electrical Engineering, National Cheng Kung University. In the military service during 1987 and 1988, he was an Associate Professor in the Institute of Resource Management, Defense Management College. In 1988, he joined the Department of Electrical Engineering, National Taiwan University, Taipei, Taiwan, R.O.C., where he is currently a Professor. During 1993–1994, he was a Visiting Consultant in the DSP Research Department, AT&T Bell Labs, Murray Hill, NJ. His current research interests are DSP architecture design, video processor design, and video coding systems.

Dr. Chen is a senior member of the Association for Computing Machinery. He is also a member of Phi Tan Phi. He received the Best Paper Awards from the R.O.C. Computer Society in 1990 and 1994. In 1991–1997, he received Long-Term Paper Awards annually. In 1992, he received the Best Paper Award at the 1992 Asia-Pacific Conference on Circuits and Systems in the VLSI design track. In 1993, he received the Annual Paper Award of the Chinese Engineers Society. In 1996, he received the Outstanding Research Award from the National Science Council and the Dragon Thesis Award.

Lilin Chen was born in Tainan, Taiwan, R.O.C., in 1966. She received the B.S. degree from National Chiao Tung University, Hsin-Chu, Taiwan, R.O.C., and the M.S. degree from Case Western Reserve University, Cleveland, OH. She is currently working toward the Ph.D. degree at National Taiwan University, Taipei, Taiwan, R.O.C.

She was an Assistant Engineer with OES/ITRI, Taiwan, R.O.C., from 1987 to 1989. She has been a Lecturer at Oriental Institute of Technology Taipei, Taiwan, R.O.C., since 1991. Her research interests include VLSI circuit design and video signal processors.