Proceedings of the 2009 IEEE
International Conference on Mechatronics and Automation
August 9-12, Changchun, China

A New Reconfigurable Logic for CNC Lathe Controllers

Jingchuan Dong+, Yunfeng Wang*, Zijing Wang+ , Taiyong Wang+

Advanced Manufacturing Technologies
and Equipments Laboratory +
Tianjin University, Tianjin, 300072, China
new_lightning@yahoo.com.cn

Department of Mechanical Engineering*
The College of New Jersey
Ewing, NJ 08534, USA
jwang@tcnj.edu

Abstract - **Reconfigurable devices have been developed to implement the control algorithms in Computer Numerically Controlled (CNC) systems recently. Reconfigurable devices provide high execution performance of CNC and flexibilities to the system design. However, little work was done on the implementation of reconfigurable logic in a CNC lathe. This paper developed a new resampling algorithm for the synchronized movement in threading and a reconfigurable logic in a CNC lathe. An experimental system was constructed and the test results validated the proposed architecture.**

Index Terms **- CNC lathe, reconfigurable logic, resample, threading**

I. Introduction

Computer Numerically Controlled (CNC) machines are widely applied in modem industries. As the control center of the machine, the CNC controller affects the machining performance in many aspects such as speed, precision and stability. With the rapid increment in machining speed and precision, the conventional software based CNC controllers are difficult to meet the computational demands. The reconfigurable hardware is introduced to solve this problem.

The reconfigurable hardware facilitates the CNC system in many ways. First, the control algorithms can be implemented by hardware with a high execution performance. The designer can take the advantage of the parallel structure of the hardware to accelerate the algorithm. In addition, the realtime execution ability of the hardware is ideal to realize the control laws efficiently. Second, the reconfigurable hardware provides flexibility for the designers. Due to the programmable characteristic of reconfigurable hardware, the designer can integrate varieties of custom circles into a single chip easily and rapidly. Third, the reconfigurable hardware provides the possibility to modify or update the hardware design in the previous version. Even if the controller has been delivered to the user, the hardware logic can still be modified to meet new requirements.

The research on the reconfigurable hardware in motion control systems have been published recently. Some work focuses on the design of position controller. Chan et al. implement the PID control algorithm on a Field Programmable Gate Array (FPGA) using a Distributed Arithmetic (DA) scheme [1]. Tao et al. append a three-grade position feedforward to the hardware PID controller [2]. Zhao et al. compares different structure of PID controller for multi-channel control [3]. Wang et al. designed a FPGA based neural network PID controller [4]. Su et al. implemented the digital difference analyzer (DDA) control algorithm on a FPGA chip [5]. Other functions in CNC controllers have also been studied. Osomio-Rios et al. implement a hardware polynomial-based profile generator with jerk limitation for CNC and robotics applications [6]. Yau et al. implemented real-time NURBS interpolation algorithm on a FPGA [7]. However, there is lack of work on implementing reconfigurable devices for the CNC lathe controller, which is the focus of this paper.

This paper developed a new resampling algorithm for the synchronized movement in threading and a reconfigurable logic in a CNC lathe. This paper will be structured as follows; In Section 2, the architecture of a CNC controller and the threading process will be discussed; In Section 3, the principle of the resampling method will be introduced; In Section 4, the hardware design of the reconfigurable logic will be described; In section 5, an example of CNC lathe controller using the proposed method will be presented.

II. Background

A. The Architecture of CNC Controller

The architecture of a CNC controller is shown in Fig. 1. The architecture includes four control layers [8]: intelligent control, motion control, device control and physical device. The Human-Machine Interface (HMI) interacts with the operator, and calls the intelligent control layer to do the machining job. In the intelligent control layer, the input NC program is interpreted into machine instructions. The machine instructions are executed by the motion control layer, which consists of an interpolator and a discrete event control module. The interpolator generates position commands according to the desired motion instructions and speed profile. Other instructions, such as tool selection and coolant control, are handled by the discrete event control module. The device control layer operates the physical devices to realize the machining process. It incorporates an axis control module and a discrete input/output control module. In the study of this paper, the device control layer is implemented by using a reconfigurable device.

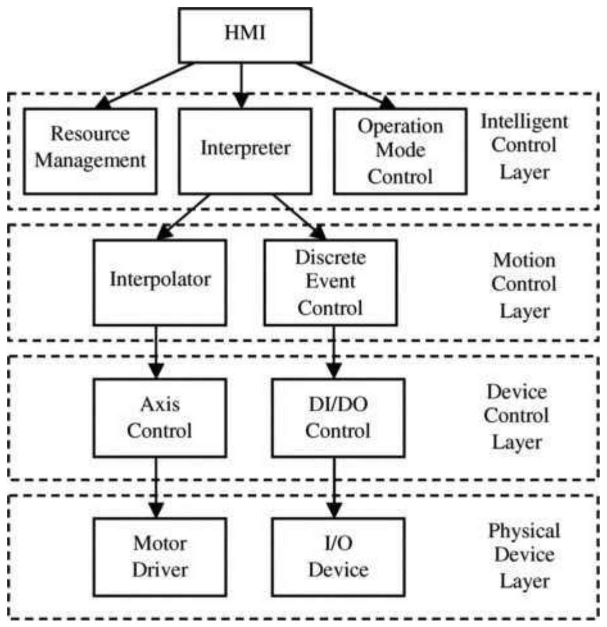


Fig. 1 Architecture of a CNC controller.

For a CNC lathe, the motion functions fall into two categories: the normal motion and the spindle synchronized motion. The normal motion is used in profile machining, while the spindle synchronized motion is used in the threading process. The principle of normal motion is similar to the motion in a milling machine, which has been successfully realized on reconfigurable devices by other researchers. This paper concentrates on the reconflgurable hardware for the spindle synchronized motion.

B. Threading on a CNC Lathe

To maintain the threading accuracy, the feed motion must synchronize with the spindle rotation as shown in (1). F is the feed speed (mm/min), *S* is the spindle speed (rpm) and *L* is the lead of the thread (mm).

$$F = SxL \tag{1}$$

An incremental rotary encoder is attached to the spindle to determine its angular position. As the spindle rotates, the encoder generates two channels (A and B) of signals that are shifted 90 degrees out of phase from each other as shown in Fig. 2. By monitoring the number of pulses and the relative phase of A and B channel, both position and rotate direction can be tracked. The encoder also provides an index signal, which pulses once per revolution, as the reference point of the angular position.

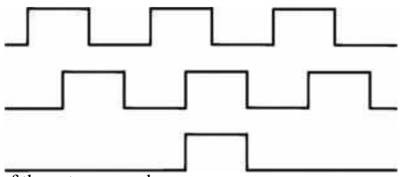


Fig. 2 Waveform of the rotary encoder.

Fig. 3 shows an example of the cutter path in threading process [9]. While the spindle rotates, the cutter moves to the points 1. Then, the controller waits for the index signal to determinate the absolute position of the spindle. Once the spindle rotates to the required position, the threading process will begin with the acceleration movement in the feed direction. During the acceleration, the thread is not precision. When the acceleration end at point 2, the feed speed arrives at the desired speed and the feed movement will synchronize with the spindle. Before the end of threading process, the cutter will begin decelerating at point 3. The thread is also not precision during the deceleration. Finally, the threading process stops at point 4 and the cutter moves away from the workpiece.

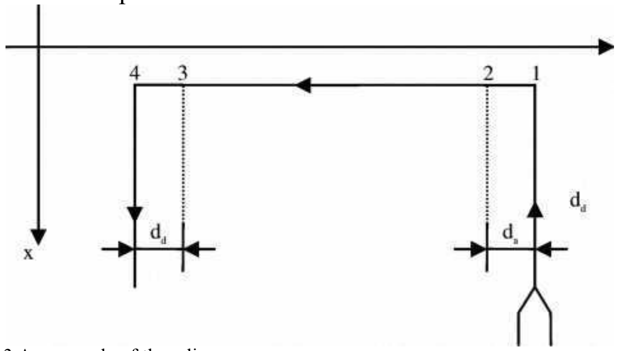


Fig. 3 An example of threading process.

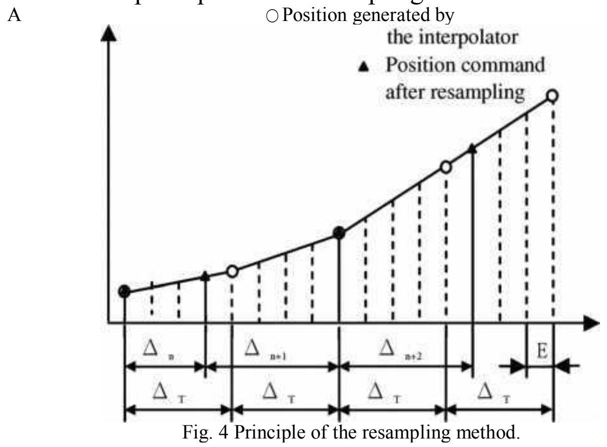
The spindle speed may fluctuate in the threading process. According to (1), the feed speed should follow to the spindle speed. The threading process may take several passes in order to limit the material removal rate or create multiple start threads. The lead may also change in a special thread. Therefore, in the threading process, the movement along the feed axis must synchronize with the angular position of the spindle. The traditional method to solve this problem can be described as following steps:

1) In each motor control cycle, sample the spindle position.
2) Call the interpolator to calculate the new cutter position according to the thread profile.
3) Update the command position for each axis.
4) Return to step 1 if the threading process is not done.

The above steps must be executed in one motor control cycle each time, thus all the steps must done in real-time. However, if the real-time execution is performed by software, the overhead of the control algorithm is high, and it is hard to achieve a short control cycle. Besides, the real-time software execution needs extra CPU time as to switch between the realtime tasks and other tasks. The proportion of the CPU time spends on the task switch will be very large if the control cycle is relatively short.

III. The Resampling Method

In our work, the resampling method is proposed to avoid the real-time execution of the interpolation algorithm when threading. Moreover, the reconfigurable hardware is employed for the motor control and other real-time functions. Fig. 4 demonstrates the principle of the resampling method:


A
○ Position generated by the interpolator
▲ Position command after resampling
Δ n
Δ n+1
Δ n+2
E
Δ T
Δ T
Δ T
Δ T
Fig. 4 Principle of the resampling method.

1) The interpolator assumes the spindle runs at a constant speed. Therefore, in each cycle, the spindle rotates a fixed degree *A(p$_T$* . Then the interpolator can sample the cutter position with the constant step size *A(p$_T$* according to the thread profile. Because the interpolator doesn't need the actual spindle position, the interpolation could be done before the real motor control cycle.

2) In the motor control cycle, the position command is generated by resampling the position curve. The position curve is reconstructed by connecting each interpolated position using straight line. Although the increment of the spindle angular position may change, the command position can be obtained by the resampling.

The position of the spindle is measured by counting the pulses from the rotary encoder. Therefore, the increment of the spindle position in each motor control cycle is the integral multiple of the pulse equivalent. We can choose

Isxp、 =*KE* ‚ (2)

where *E* is the pulse equivalent of the spindle encoder and ^ is a positive integer. Assuming at the two successive interpolating times the spindle positions are *q)$_n$* and *(p*$_{n+x}$, and the correspondent interpolated positions for the feed axis are A_n and A_{n+1}, since the position curve is reconstructed by linear function, the position function in the interval can be written as

(3)

$= A_{n+1} - A_n$

As the spindle position is represent by the counts of encoder pulses, the *(p* in (3) can only take discrete values. Let

g> = g)$_m$=g)$_n$+mE， *m* = . (4)

Substituting (2) and (4) into (3), we obtain

^((P$_m$) = A — = (5)

K

Equation (5) can be further written in incremental form

j0p$_{m+}$i) = 40p$_m$) + ^4$_m$
< ^4 , *m* = *0, l--K.*
(6)

K$^=$T

T = *K*

Equation (6) is the algorithm [8] for the resampling method. The algorithm involves addition and division. In the calculation, if *AA* is a fixed-point number and

尤=2",(7\^ is a non-negative integer), (7)

the division operation can be realized by the shifting operation. Therefore, the resampling algorithm is very simple and it is suitable to be implemented by hardware.

By assuming the spindle rotates at a constant speed, the equivalent interpolation cycle *7]* can be determined as follows

60 – g TV 60 where *P* is the number of generated pulses per revolution of the spindle. From (8) we can see that interpolation cycle is independent of the motor control cycle. Choosing a larger *N* allows a longer equivalent interpolation cycle, so the computation load can be reduced. However, a large *N* may also reduce the resampling precision. So the balance between computation cost and precision should be considered when selecting *N*.

IV. The Reconfigurable logic

Many of the CNC tasks can be realized either by hardware or by software. With the development of programmable logic devices (PLD), such as Field Programmable Gate Array (FPGA) and Complex Programmable Logic Device (CPLD), more and more control functions can be accomplished by hardware. The parallel nature of hardware allows the control algorithms to be run in real time. Moreover, the reconfigurable ability of PLD chips provides a flexible hardware platform which can be developed and modified rapidly.

A. Architecture of the Reconfigurable Logic

The structure of reconfigurable logic design in this work is shown in Fig. 5. The structure consists of several reconfigurable modules. For each feed axis, there are a axis buffer, resampler, command position multiplexer and a motor controller. To simplify the design, only the increment of position is utilized for the transport of position data. This method avoids using the absolute position, thus saves the bit lines in the data exchange among different modules. Each axis contains two data paths for the position data. The data path is selected by the command position multiplexer according to the control mode. In the normal mode, the resampler is bypassed and the command position for the motion controller is provided by the axis buffer. In the threading mode, the resampler is applied for the resampling of the position curve.

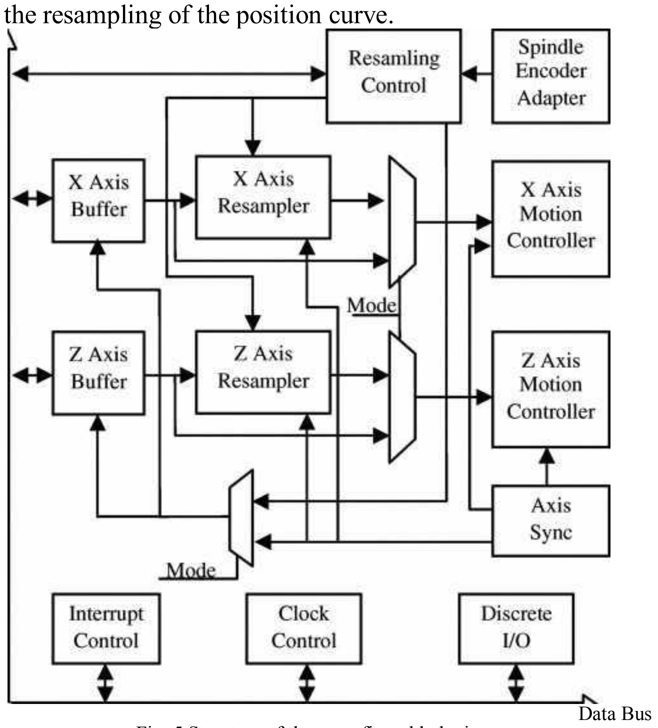


Fig. 5 Structure of the reconfigurable logic.

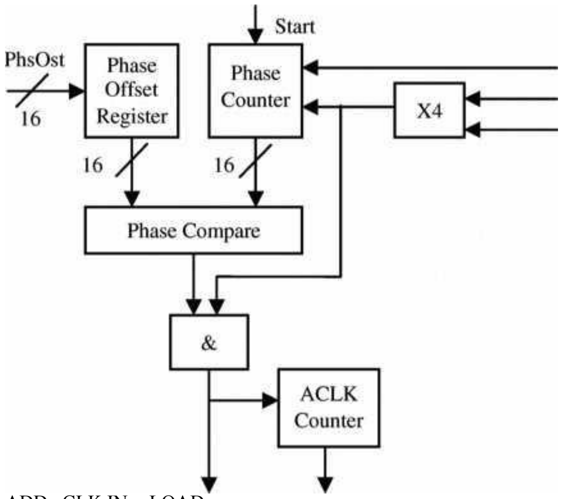
The axis buffer keeps the next position increment given by the interpolator. Therefore, the next interpolated position can be loaded from the axis buffer immediately. Then, the new interpolated position can be store to the buffer again. The resampler performs the resampling algorithm. The resamplers in the two channels are controlled by a resampling control module. The resampling controller controls the resampling process according to the spindle position. The motion controller implements the position control algorithm and provides interface to the motor driver. Several control algorithms are available, such as Digital Differential Analyzer (DDA), PID and Fuzzy controller. In this study, the DDA controller is employed. The two motion controllers are synchronized by the control signal from the axis synchronization module.

Some complementary modules are also incorporated in the PLD. The spindle encoder adapter is the interface to the spindle encoder. Digital filters are integrated in the adapter to prevent the noise influence. The discrete I/O module handles the general purpose input and output signals of the CNC controller. The interrupt control module is used to generate interrupt signal to the CPU when certain event occurs. The clock control module provides the clock signals to other modules.

The hardware implementation of motion controller and other complementary modules have been comprehensively discussed in literatures, the rest of this paper will discuss on the realization of the resampling algorithm.

B. Structure of the Resampling Logic

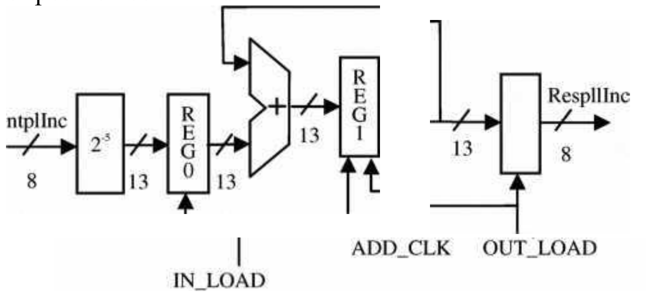
Fig. 6 shows the structure of the resampling controller. The signals from the spindle encoder adaptor are used to determine the position of the spindle. To get a high resolution, the input pulse frequency is quadrupled. Before the threading starts, the phase offset is loaded to the phase offset register and the output of phase compare is low. Therefore, the output of the AND gate is low. When the start signal arrives, the phase counter will wait for the index signal. After the coming of the index signal, the phase counter start to count the input pulses. When the pulses count match the phase offset, the output of phase compare will goes high. The quadrupled encoder input signal will go through the AND gate as the clock for the addition operation. The resampler will perform the addition of (6) when the ADD CLK signal comes. The ACLK counter is a A^-bit counter for counting the addition times. It takes *K* addition times to overflow. When the overflow occurs, the IN LOAD signal is set and the resampler will load the new interpolated position form the axis buffer.



ADD –CLK IN –LOAD

Fig. 6 Structure of the resampling controller.

The structure of the resampler is shown in Fig. 7. It comprises of one shifter, one adder and three registers. The shifter is employed to perform the division operation in (6). In this work, we choose *N^5* and the input AA is represented by an 8-bit fixed point number. The output of the resampler is also an 8-bit fixed point number. The AA_m is obtained by the TV-bit right shift of AA. When the IN LOAD signal comes, the new AA_m is latched in the REGO. The 13>bit adder is used to do the addition operation in (6). Every time the ADD CLK signal comes, REG1 latches the result of the addition. If the motor controller requires the increment of next command position, the OUT LOAD signal will goes high. Then, the resampled position, i.e. the higher 8-bits value of REG1, is latched to REG2. The OUT LOAD signal also clears the higher 8-bits of REG 1. Therefore, the value in REG1 is the increment of position command.

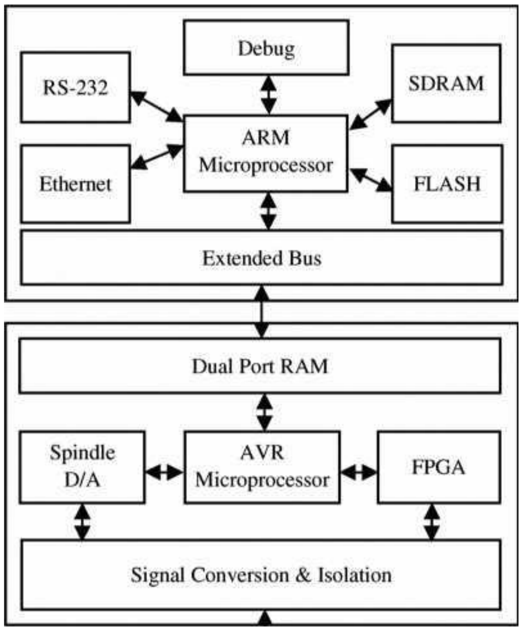


V. Experimental Verification

Fig. 7 Structure of the resampler.

A CNC lathe controller was built to test the proposed control method. The diagram of the experimental CNC system is shown in Fig. 8. The system consists of two parts: the nonreal-time part and the real-time one. The non-real-time part includes a 32-bit ARM 920T microprocessor, a flash memory, a SDRAM, communication ports and a debug unit. The nonreal-time part runs the Linux operating system. The control software includes the user interface, interpreter, interpolator, sequential logic, communication, NC file management and other functions. The hardware of the real-time part includes a 8-bit AVR microcontroller, a dual port RAM, a FPGA chip, a D/A converter and the circuits for signal conversion and isolation. The dual port RAM is used to buffer the real-time control commands, which includes the interpolated positions. The AVR microprocessor was used to fetch the control commands and send them to the FPGA and spindle D/A converter in real-time. The reconflgurable control logic was implemented on the FPGA. The D/A spindle converter controls the spindle speed. Fig. 9 shows the hardware setup of the CNC controller.

In the experimental system, the resolution of the spindle encoder was 4800 pulses per revolution (after quadruple). The motor control cycle is 1.02 milliseconds. Several machining programs were tested on this controller to examine the proposed reconflgurable logic. A profiling program tested the straight and arc feed functions in the normal control mode, while three other programs verified the resampling algorithm in the threading mode. The parameters of the threads are listed in table I. In the threading tests, the spindle speed was programmed to 500 rpm, and the actual speed was between 486.4 to 518.5 rpm. According to (8), the equivalent interpolation cycle in the test was 0.8 milliseconds. Fig. 10 shows the threading process in the test. Fig. 11 shows the finished parts. All of the test programs worked as expected. The results show that the proposed architecture is feasible to the CNC lathe controller.



^ To the CNC lathe Fig. 8 Diagram of the experimental system.

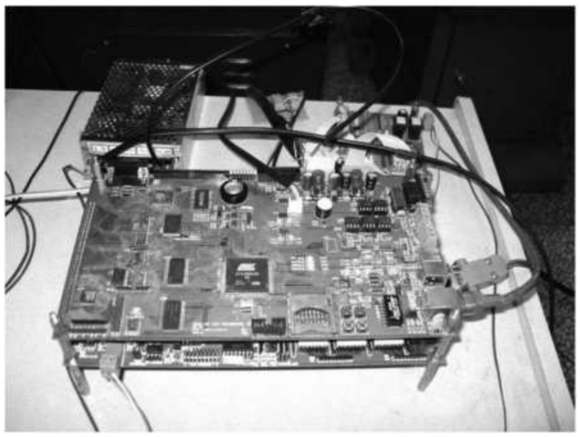


Fig. 9 Hardware setup of the controller. TABLE I

PARAMETERS OF THE TEST THREADS

Parameter	Test 1	Test 2	Test 3
Major diameter (mm)	20	22	16-22
Minor diameter (mm)	18.4	20.4	14.4-20.4
Length (mm)	30	30	30
Lead (mm)	1.5	3	1.5
Pitch (mm)	1.5	1.5	1.5
Taper rate	0	0	0.2
Comment	Single-start straight thread	Double-start straight thread	Single-start tapered thread

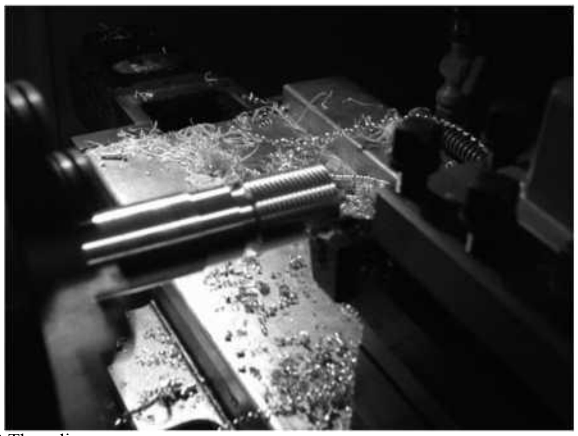


Fig. 10 Threading process.

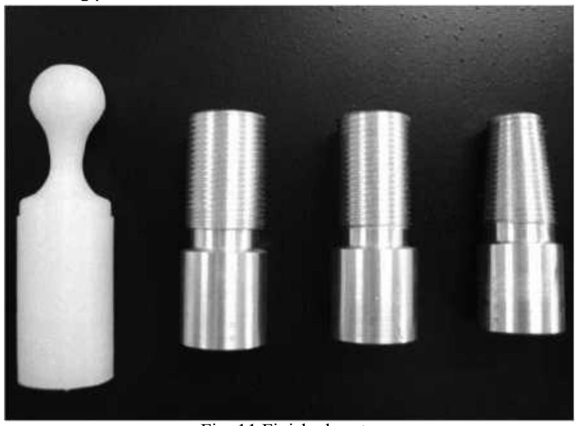


Fig. 11 Finished parts.

It is notable that, by utilizing the reconfigurable logic, the motor control task and resampling algorithm could be done by hardware. The reconfigurable hardware enhances the CNC system performance. Moreover, this architecture allows the interpolated points to be calculated before the threading, thus eliminates the need to run the interpolator in real-time. By running interpolation in a batch mode, the overhead in the real-time context switch is reduced, and a higher data throughput is possible. In this experiment, the control software is running on a standard Linux system, which is a non-realtime operating system. Thereby, developers can use the standard programming tools to build the CNC control software.

VI. Conclusion

This paper presents the development of reconfigurable hardware architecture for the CNC lathe controller. We proposed a new resampling algorithm for the threading process of CNC lathes. This resampling algorithm can maintain the threading accuracy despite the fluctuation of the spindle speed. By utilizing the resampling algorithm, the interpolation task and motor control task can run asynchronously. Thereby, the interpolation can be done in a non-real-time manner. This will simplify the software design and provide a higher data throughput.

From the designer's view, the resampling algorithm is suitable for the implementation by hardware. The structure of the resampling logic is also presented in this paper. The resampler, along with other reconfigurable logic structures, can be easily integrated into a programmable logic device. The application of reconfigurable hardware enhances the system performance and provides flexibility for system design. The experiment has successfully demonstrated the feasibility of the proposed resampling algorithm and the corresponding reconfigurable logic.

Acknowledgment

This research was sponsored by the National High-Tech. R&D Program, China (No. 2007AA042005), and the Research Fund for Doctoral Program of Higher Education, China (No. 20060056016).

References

[1] Y.F. Chan, M. Moallem and W. Wang, "Efficient Implementation of PID Control Algorithm using FPGA Technology''.岑 / 五 五 五 ○ « *Decision and Control,* Dec. 2004, pp. 4885-4890.

[2] Y.D. Tao, H. Lin, X.H. Zhang and Z.C. Wang. "Efficient Implementation of CNC Position Controller using FPGA^{,5}. *The IEEE International Conference on Industrial Informatics,* July 13-16, 2008, pp.l 177-1182.

[3] W. Zhao, B.H. Kim, A.C. Larson and R.M. Voyles. ^{U}FPGA Implementation of Closed-Loop Control System for Small-Scale Robot". *12th International Conference on Advanced Robotics,* July 18-20, 2005, pp.70-77.

[4] J.Z. Wang, Y.P. Chen, J.M. Xie, B. Chen and Z.D. Zhou. ^{C}TPGA based Neural Network PID Controller for Line-scan Camera in Sensorless Environmenf^{5}. *Fourth International Conference on Natural Computation,* pp. 157-161.

[5] K.H. Su, C.K. Hu and M.Y. Cheng, "Design and Implementation of an FPGA-based Motion Command Generation Chip". 瓯五*Conference on Systems, Man and Cybernetics,* Oct. 8-11, 2006.

[6] R.A. Osomio-Rios, R.J. Romero-Troncoso, G. Herrera-Ruiz and R. Castaneda-Miranda. "FPGA Implementation of Higher Degree Polynomial Acceleration Profiles for Peak Jerk Reduction in Servomotors". *Robotics and Computer-Integrated Manufacturing,* Vol. 25, 2009, pp. 379-392.

[7] H.T. Yau, M.T. Lin and M.S. Tsai. "Real-time NURBS Interpolation using FPGA for High Speed Motion Contror^{5}. *Computer-Aided Design,* Vol. 38, 2006, pp. 1123-1133.

[8] S. Park, S.H. Kim and H. Cho. "Kernel Software for Efficiently Building, Re-configuring, and Distributing an Open CNC Controller". / 咐 J dafv *ManufTechnol,* Vol. 27, 2006, pp. 788-796.

[9] Beijing-Fanuc Mechatronics Corporation, *FANUC Series Oi-TC Operator's Manual (in Chinese),* Beijing, China, 2004.