Advances in Intelligent Systems Research (AISR), volume 151
2018 International Conference on Computer Modeling, Simulation and Algorithm (CMSA 2018)

The Development of Multi-axis Embedded CNC System Based on Dual-core Chip

Xu Liu[1], Haixin Zou[2], Xiangfa Kong[3,*] and Jun Lu[4]
[1]Shenzhen youqi lighting co. LTD. Shenzhen, 518035, China
[2]Shenzhen Institute of Information Technology, Shenzhen 518172
[3]Engineering Training Center, School of Mechanical and Automotive Engineering, South China University of Technology, Guangzhou 510641, China
[4]Guangzhou Institutes of Biomedicine and Health (GIBH), Chinese Academy of Sciences, Guangzhou, 510530, China
[*]Corresponding author

***Abstract*—At present, from the selection of the chip to the molding of the product, the embedded CNC system will encounter various problems, such as insufficient performance of the processor, hidden dangers of the physical communication scheme between the chips, insufficient reconfigurability of the platform, and transmission rate limitation and so on. This makes the development cycle of the embedded numerical control system very long. This paper combines the rapid development and mature multi-core technology of electronic products, and develops an open multi-core embedded CNC system platform based on high-performance TMS320DM8148. The overall structure of the platform is based on a multi-core CNC system, including hardware platform construction and circuit design, dual-core and inter-core communication, and dual-core and FPGA high-speed bus communication. The reconfigurability developed by FPGA is used to develop each module in the CNC system and achieve many Axis servo driver control, and finally gives the system startup and control process, the firmware program is written to the dual-core CNC system for actual processing experiments. From the results of the performance test of the CNC system, the indicators could meet or exceed the performance requirements of the existing high-performance CNC systems, and also have open, modular and network control functions. The system platform has been successfully applied Five-axis linkage CNC machine tools, CNC milling machine, CNC engraving machine, all-electric servo CNC bending machine, spring machine and other types of CNC machine tools, reducing the design difficulty, greatly reducing the development cycle, allowing users to focus on solving specific applications Problems, and it thus quickly promote the development of the national economy and the overall level of industrial manufacturing, and has produced a very good economic and social benefits.**

Keywords—embedded; dual-core communication; multi-axis linkage; FPGA interpolation; CNC system

I. INTRODUCTION

Under the influence of the economic globalization and the information technology revolution, the international manufacturing industry is undergoing profound changes, and the scale and level of the manufacturing industry has become an important symbol to measure the comprehensive strength of a country. Numerical control (NC) machine tool, especially high grade CNC machine tool, is a hot field of international equipment manufacturing competition. China has put the high grade CNC machine tools and basic manufacturing equipment into the national science and technology major special projects. [1]

Because of backward development of numerical control system, the weak strength of research and development team, and not enough investment, for a long time, CNC products are always rapid expansion in the low end, have slow progress in the middle end, and rely on imports of high-end [2].

1). The high-end products are monopolized by the international numerical control giants, and the high performance machine tools and their related key technologies are not accessible to China [3]

2). The CNC system of most manufacturers in China is based on PC type CNC system, which has a large number of redundant hardware and software, low real-time performance, low stability and high power consumption. The research of embedded CNC system has become a new direction for the development of CNC system at home and abroad.

3). There are a large number of domestic researches on embedded CNC system, but most of those are for low-cost embedded system for low-end market. Their hardware structure is simple low computing power, and there is a problem of real-time communication delay using the discrete chip design.

At present, during the equipment manufacturing industry high-speed development, the importance of numerical control machine tool is growing, and the demand for high-end CNC system is bigger and bigger, however, domestic high-end CNC system market is basically occupied by foreign manufacturers. Therefore, it has become a top priority to strengthen the basis research and the key problem research in the equipment manufacturing industry development in our country. It is of great significance to promote China's high-end CNC system independent design and development ability and international competitiveness. [4]

Aiming at the existence of the above questions, this paper proposed a new high-performance embedded CNC system hardware solution using TMS320DM8148, a heterogeneous dual-core processor, and FPGA chip as a numerical control system of the operation unit and the control core. TMS320DM8148 chip is embedded ARM Cortex-A8 kernel

and DSP C674x kernel. Through two chips of powerful operation ability and rich hardware resources, the high-performance embedded NC system with completely independent intellectual property rights was designed.

II. METHODS

A. Hardware Platform Construction. [5]

TMS320DM8148 contains the cortex-a8 processor kernel with a main frequency of 1GHz and an 800M C674x kernel, and a large number of peripherals. With TMS320DM8148 as the hardware platform to design dual-core communication, it has the following advantages: good openness; good programming; low power consumption; the two cores run independently and separately after the system is started.

The address of peripherals and memory of the dual core processor, DSP and ARM, are unified together, they shared memory space and data interaction.

As shown in Figure I, a CNC system based on TMS320DM8148 dual-core chip included human-computer interaction interface, TMS320DM8148 microprocessor, USB interface, Ethernet interface, Micro SD interface, FPGA and peripheral circuit; and a human-computer interaction interface.

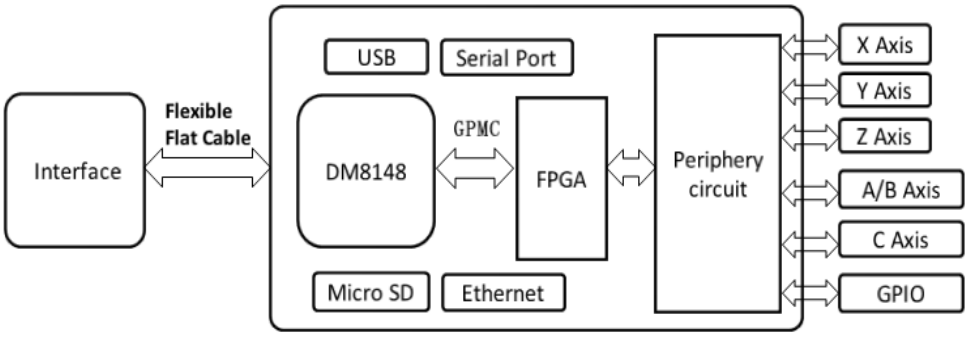

FIGURE I. THE STRUCTURE BLOCK DIAGRAM OF THE CNC SYSTEM

TMS320DM8148's ARM core is mainly used for multi-task scheduling: man-machine interface control and data exchange, interface real-time refresh, CNC G command pre-reading and analysis, while DSP core is mainly responsible for motion control operations: regional division, regional forward, speed planning, regional motion control pre-calculation, smoothing processing and interpolation operation; dual-core processor transmits interpolation and other motion control data to the FPGA, and then control multi-axis synchronous interpolation through the peripheral circuit to realize multi-axis linkage of CNC machine tools.

B. Dual-core Chip Inter-Processor Communication [6-8]

The multi-core system is composed of ARM+DSP, which is called heterogeneous multi-core (isomorphism means the structure of the internal core is the same, and heterogeneous refers to the internal nuclear structure is different) in order to provide efficient heterogeneous multi-core cooperation for a heterogeneous multi-core processor, it is necessary to establish a heterogeneous multi-nuclear communication mechanism. The dual-core system uses the SYSLINK driver to design a handshake mechanism that can be used to communicate between DSP and ARM. The ARM side runs the Linux operating system for human-computer interaction; the DSP side runs the SYSBIOS operating system to perform data storage and buy-time operations. Both use SYSLINK for multi-core communication and call SYSLINK API to complete the communication between ARM and DSP.

In the ARM Linux operating system of the TMS320DM8148, SYSLINK provides a "slaveloader" component to load, start, and stop the DSP processor, designs the management of the DSP core, and also uses the "slaveloader" component to run the SYSLINK sample program. The compiled application program is downloaded to the development board, and the run.sh script is written after running, which realizes the dual-core communication process. The contents of the script are:

```
/*********************************************/
set-x
./slaveloader startup DSP server_dsp.xe674
./app_host DSP
./slaveloader shutdown DSP
/*********************************************/
```

The basic flow is: The ARM side starts the DSP and loads the .xe674 or the out format SYS/BIOS file → Starts the ARM side application → Turns off the DSP core.

When the user develops the application, you can customize the application name, such as start CNC_FiveAxis.out:

```
/*****************************************/
./slaveloader startup DSP CNC_FiveAxis.out
/*****************************************/
```

C. Dual-core and FPGA Communication [9-10]

The dual-core system drives the GPMC interface by calling the GPMC driver function under Linux to achieve high-speed data communication with the FPGA. GPMC's full name is General-Purpose Memory Controller, which is a universal

memory controller. It is an interface used by TI's chips to communicate with external memory devices such as NOR FLASH, NAND FLASH, SRAM, and so on.

The GPMC basic programming model offers maximum flexibility to support various access protocols for each of the six configurable chip-selects. Use optimal chip-select settings, based on the characteristics of the external device:

1) Different protocols can be selected to support generic asynchronous or synchronous random-access devices (NOR flash, SRAM) or to support specific NAND devices.

2) The address and the data bus can be multiplexed on the same external bus.

3) Read and write access can be independently defined as asynchronous or synchronous.

4) System requests (byte, 16-bit word, burst) are performed through single or multiple accesses. External access profiles (single, multiple with optimized burst length, native- or emulated-wrap) are based on external device characteristics (supported protocol, bus width, data buffer size, native-wrap support).

5) System burst read or write requests are synchronous-burst (multiple-read or multiple-write). When neither burst nor page mode is supported by external memory or ASIC devices, system burst read or write requests are translated to successive single synchronous or asynchronous accesses (single reads or single writes). 8-bit wide devices are supported only in single synchronous or single asynchronous read or write mode.

6) To simulate a programmable internal-wait state, an external wait pin can be monitored to dynamically control external access at the beginning (initial access time) of and during a burst access.

The system hardware connection mode configures the GPMC interface as asynchronous mode and sets the NOR FLASH and non-address data line multiplexing modes to communicate with the FPGA, as shown in Figure II.

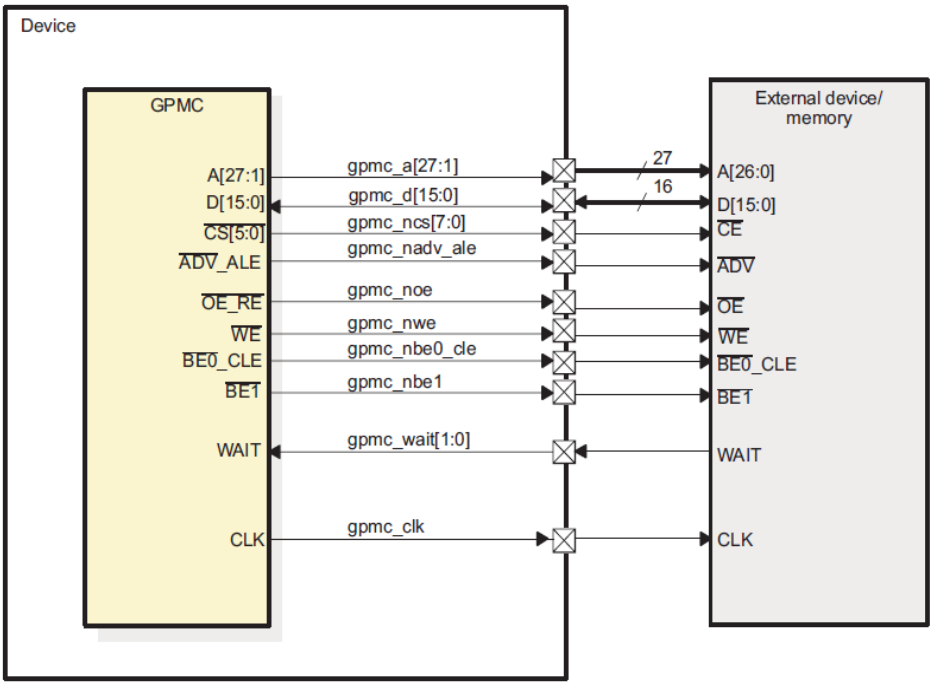

FIGURE II. GPMC TO 16-BIT NONMULTIPLEXED MEMORY

1) GPMC memory allocation, the following code for reference

```
/*********************************************
/

const int MMU_BASE = 0x08010000;

volatile int *mmu_sysconfig = (int *) (MMU_BASE +
0x10);

volatile int *mmu_sysstatus = (int *) (MMU_BASE +
0x14);

*mmu_sysconfig = 0x00000002; // execute software reset

while ((*mmu_sysstatus & 0x00000001) != 0) // wait for
reset

//0x1000000 address virtual to 0x11000000, base address
0x11000000.

mmuWriteStaticTlb(entry, 0x11000000, 0x01000000);

/*********************************************
/
```

2) There are gpmc.h and gpmc.c files in the Linux kernel source code. The functions inside support GPMC are very comprehensive. To use these functions, you need to first call omap_init_gpmc() and omap_init_elm() to initialize the GPMC

device, otherwise you cannot call insmod to load the drive module.

The key to the GPMC driver design is to properly configure the relevant registers, otherwise the desired operating results will not be obtained. For NOR type devices, simply configure the GPMC_CONFIG1~GPMC_CONFIG7 registers.

The Linux gpmc driver source code is in /arch/arm/mach-omap2/gpmc.c. The configuration of the GPMC related seven special registers is as follows:

```
/*********************************************/
GPMC_CONFIG1_1 = 0x00001010;
GPMC_CONFIG2_1 = 0x00101080;
GPMC_CONFIG3_1 = 0x00020201;
GPMC_CONFIG4_1 = 0x0f031003;
GPMC_CONFIG5_1 = 0x000f1111;
GPMC_CONFIG6_1 = 0x0f030080;
GPMC_CONFIG7_1 = 0x00000F42;
gpmc_write_reg(GPMC_IRQENABLE, 0);
gpmc_write_reg(GPMC_TIMEOUT_CONTROL, 0);
gpmc_write_reg(GPMC_CONFIG, 0);// must be configured here
//1 LIMITEDADDRESS 0-1 Limited Address device support
gpmc_cs_write_reg(GPMC_CS, GPMC_CS_CONFIG1, gpmc_nor[0]);
gpmc_cs_write_reg(GPMC_CS, GPMC_CS_CONFIG2, gpmc_nor[1]);
gpmc_cs_write_reg(GPMC_CS, GPMC_CS_CONFIG3, gpmc_nor[2]);
gpmc_cs_write_reg(GPMC_CS, GPMC_CS_CONFIG4, gpmc_nor[3]);
gpmc_cs_write_reg(GPMC_CS, GPMC_CS_CONFIG5, gpmc_nor[4]);
gpmc_cs_write_reg(GPMC_CS, GPMC_CS_CONFIG6, gpmc_nor[5]);
/*********************************************
/
```

3) After the configuration is complete, modify the config file (first copy /arch/arm/configs/ti8168_evm_defconfig to the upper arm folder) and load the gpmc file into the kernel.

```
/*********************************************
/
CONFIG_GPMC_FPGA=y // Compile to kernel
CONFIG_GPMC_FPGA=m //
/*********************************************/
```

4) Compile the kernel and load the GPMC driver module: insmod fpga.ko.

D. FPGA Expansion Module Design [11-13]

The system uses Intel Cyclone10 series FPGAs. Compared with previous generations of Cyclone FPGAs, the Intel Cyclone10 series could save more cost and power, provide low static power consumption, and cost-optimized features.

The control expansion of the digital control system FPGA generally includes a motor control output module (Motor control output module), a motor status feedback module (Encoder feedback module), a general-purpose input/output module (GPIO module), AD/ The AD module (AD/DA module) is shown in Figure III.

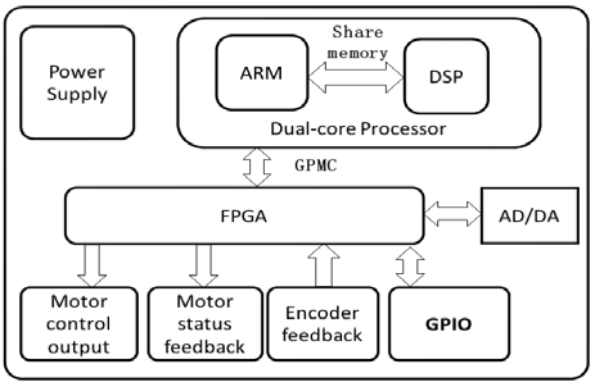

FIGURE III. FPGA EXTENSION BLOCK DIAGRAM

Motor control output module is mainly used for multi-axis pulse synchronous interpolation output. Motor status feedback module mainly feedback the status of the motor, such as positioning operation, positioning completed, alarm and warning, Encoder feedback module is mainly used to feedback the current absolute coordinates of the motor, GPIO module is an ordinary input and output port, such as lubrication and cleaning fluid. AD/DA module is a variety of analog input signals and control output of CNC machine tools, such as temperature detection and liquid level detection. The following would focus on the Motor control output module.

Motor control output module is used for multi-axis linked pulse interpolation output. Its pulse interface mainly includes two parts: PWM module and LPM module.

As shown in Figure IV, the PWM module (PWM.VHD) is used to generate programmable frequency pulses and control the speed. The LPM module (LPM.VHD) is used to output a programmable number of pulses and the rising edge is valid.

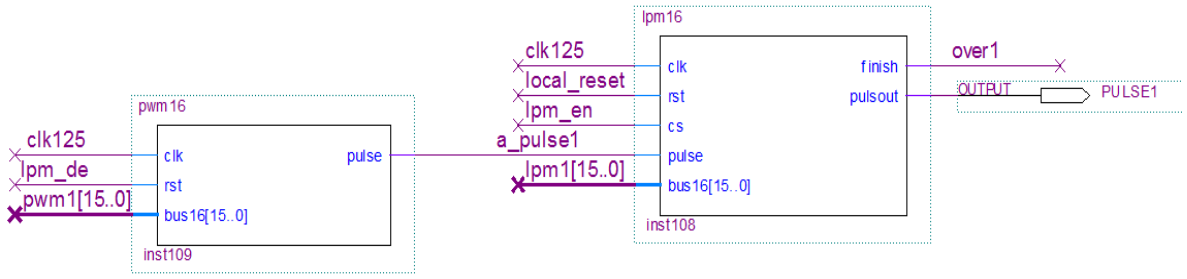

FIGURE IV. PULSE INTERFACE OF MOTOR CONTROL OUTPUT MODULE

The OVER signal is 1 to indicate that the current interpolation pulse in the LPM is finished, and the channel is closed, and the next set of interpolation pulses can be written. When each axis OVER signal is 1, the output causes the DSP core to generate an interrupt.

The pulse output processing is shown in Figure V: (Tc is the interpolation period)

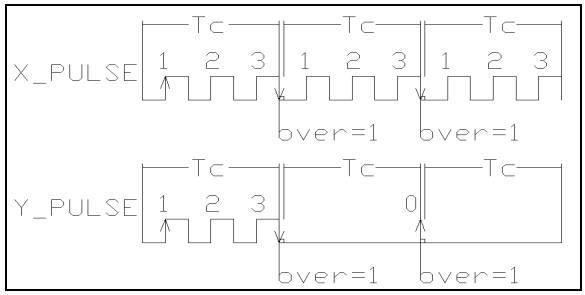

FIGURE V. PULSE OUTPUT PROCESSING.

If the interpolation pulse is not 0, and it would be written into LPM. When LPM value is written as 3, and the count value at the rising edge is incremented by 1; OVER is set at the falling edge after 3 counts are completed, and it would recount again; the next small area is the software calculation time.

If the interpolation pulse is 0, the processing in LPM is different from non-zero. It is different from non-zero. It is set to OVER at the rising edge and thus writes 0 to PWM. It needs to make the half cycle time and interpolation time the same.

E. System Startup and Control Process

The embedded linux operating system is started. There are three system files: bootloader (uboot), kernel (uImage), and root file system (rootfs). After the system is powered on, it is started in the order of uboot->kernel->rootfs. Since the development board has a variety of storage media, three files can be placed on any media that can be stored, thus leading to a variety of file startup methods. Generally the development board will have flash (Nor or NAND), mmc, emmc, sd card and so on. System files can be programmed in any of these. In the development process, it is often necessary to change the kernel or modify the application program. If the memory medium on the board is reprogrammed after each modification, it may be troublesome. Therefore, to facilitate debugging, uImage and rootfs can also be started from the network, i.e. nfs startup. However, uboot can only boot from the on-board media. The startup process is actually to copy the file to be started from the storage location to the memory space and then run it in the memory. Therefore, the so-called different locations start, that is, copy from different locations. The uboot image file needs to be created first. The kernel kernel image file is mirrored with the rootfs file system and burned into the target board's storage medium. The production process is the same as the general embedded linux development and will not be repeated.

Figure VI shows the dual-core chip and FPGA communication block diagram. As shown in Figure VI, after the system is started, shared memory access is performed between the dual cores through interrupts. Dual cores communicate with the FPGA via the GPMC interface.

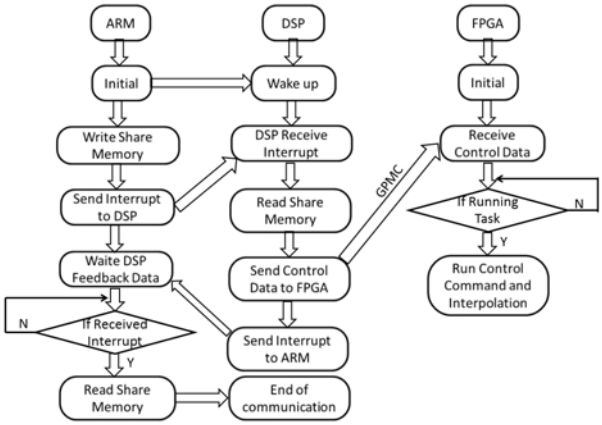

FIGURE VI. BLOCK DIAGRAM OF DUAL-CORE CHIP AND FPGA

Here need to pay attention to the correct configuration of the interrupt. The following main codes are for reference.

```
/*********************************************
/

const int GPIO0_BASE = 0x48032000;

volatile int *GPIO0_sysconfig = (int *) (GPIO0_BASE + 0x10);

volatile int *GPIO0_SYSSTATUS = (int *) (GPIO0_BASE + 0x114);

volatile int *GPIO0_IRQSTATUS_SET_0 = (int *) (GPIO0_BASE + 0x34);

volatile int *GPIO0_RISINGDETECT = (int *) (GPIO0_BASE + 0x148);
```

```
volatile int *GPIO0_IRQSTATUS_CLR_0 = (int *) (GPIO0_BASE + 0x3C);
*GPIO0_sysconfig = 0x00000002; // execute software reset
while ((*GPIO0_SYSSTATUS & 0x00000001) != 0) // wait for reset
int GPIO0_IRQ_BIT_SET0 = 0;
GPIO0_IRQ_BIT_SET0 = 0x1<<24;//gpio0 bit 24(0start)
*GPIO0_IRQSTATUS_SET_0=GPIO0_IRQ_BIT_SET0;
*GPIO0_RISINGDETECT =GPIO0_IRQ_BIT_SET0; //falling edge
*GPIO0_IRQSTATUS_CLR_0 = 0;
/*********************************************/
```

Figure VII shows the control flow of a dual-core chip CNC system.

1) CNC system power on, dual-core processor ARM core initialization, embedded linux operating system starts, while waking up the DSP core, the next FPGA initialization, human-machine interface display is shown normally.

2) ARM kernel linux operating system for multi-task scheduling: human-computer interface control and data interaction, interface real-time refresh, CNC G command pre-reading and analysis, write shared memory, send interrupt to the DSP core. The microprocessor analyzes the G-commands of the CNC and obtains the coordinates and setting speed of the space where the motors need to move.

3) The DSP core receives ARM interrupts, reads shared memory, and receives motion code data; performs motion control operations: zone division, zone look-ahead, speed planning, zone motion control pre-calculation, smoothing processing, and interpolation operations. The dual-core microprocessor uses a space curve trajectory interpolation method to interpolate several subdivision points between adjacent coordinate points to generate the position, velocity, acceleration, and motion of each motor required for each interpolation cycle. The jerk, etc., is stored in the interpolation buffer of the system software program.

4) Dual-core microprocessor interacts with FPGA data: DSP performs interpolation output, FPGA receives interpolation data transmitted from DSP interpolation buffer, performs multi-axis synchronous interpolation, and realizes multi-axis linkage; FPGA synchronously reads peripheral servo in real time The motor status feedback value and encoder feedback value are transmitted to the dual-core microprocessor through the parallel bus.

5) DSP sends an interrupt to ARM, ARM reads the shared memory, and then displays the servo motor status feedback value and encoder feedback value on the interface of human-computer interaction machine.

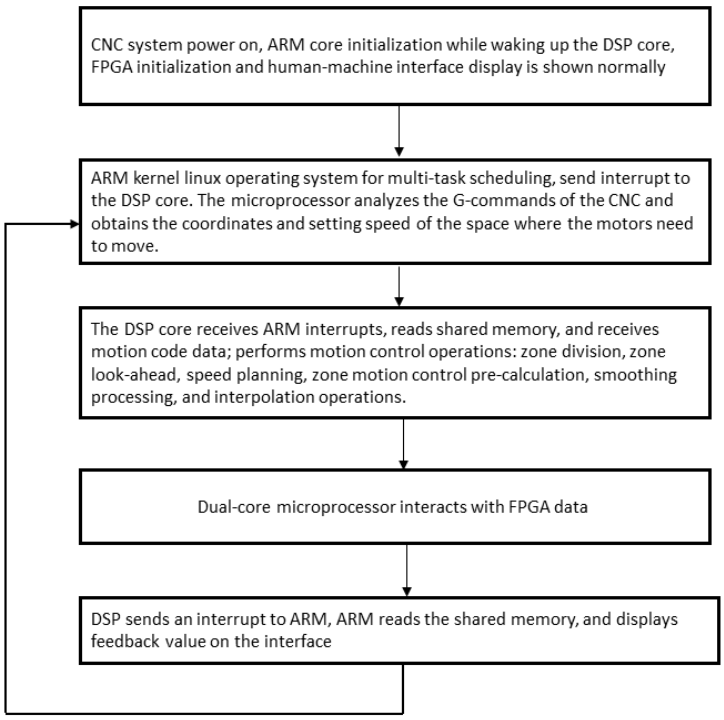

FIGURE VII. CONTROL FLOW OF DUAL-CORE CHIP CNC SYSTEM

III. APPLICATION AND RESULT

The system had also been successfully applied to five-axis linkage CNC machine tools, CNC milling machines, CNC engraving machines, all-electric servo CNC bending machines, spring machines and other types of CNC machine tools. The machine tool has operated stably and reliably, satisfied the machining accuracy requirements, and improved the machining efficiency.

Figure VIII shows the effect of machining a small shell on a CNC engraving machine. Set the feed speed to F=3000, the spindle speed to S=20000, and the machining time to 3 minutes. The results show that the knife pattern is clear, the surface finish is good, there is no broaching marks, and the overall processing effect is good.

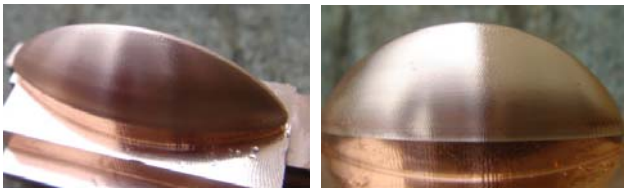

FIGURE VIII THE EFFECT OF MACHINING A SMALL SHELL

IV. CONCLUSIONS

This paper adopted TMS320DM8148 heterogeneous dual-core processor + FPGA as the hardware platform to design an embedded CNC system with intellectual property rights. The system has the advantages of good man-machine interface, abundant interface resources, good stability and high computing capacity. The design of the overall hardware architecture of the embedded CNC system and the communication between the TMS320DM8148 core and the communication with the FPGA bus were studied. Finally, the function and feasibility of the embedded CNC system were verified through experiments. The embedded CNC system has been successfully applied to five-axis linkage CNC machine tools, CNC milling machine, CNC engraving machine, all-electric servo CNC bending machine, spring machine and many other types. CNC machine tools have produced very good economic and social benefits.

This paper has not introduced in detail in the concrete circuit realization of each hardware module of the whole system, system memory allotment, system start, ARM kernel application program development, FPGA module VHDL language realization, DSP motion control process software code implementation, etc. And experts can contact me by email to discuss.

ACKNOWLEDGEMENTS

This work is supported by (1) Guangzhou NLKER Precision Machine Tool co., Ltd; Shenzhen youqi lighting co. LTD.

REFERENCES

[1] Li Xiaodong, Qian Sisi. “Numerical Control System: State of Art and Trends" Cai Ruilong, Mechanical Science and Technology for Aerospace Engineering, 2016(03)

[2] WANG Xiwen. “Industry 4.0, Internet+, Made in China 2025 The Future Direction of China's Manufacturing Transformation and Upgrading ". National Governance Weekly, 2015, (23):12-19.

[3] Wang ying. “Countermeasures and Suggestions for the development of China's numerical control machine tool industry". LAN tai world. 2014(S5)

[4] ZHOU Ji, SHAO Xiny, ZHOU Yanhong. “Strategic Significance and Technical Route of "NC Generation" Mechanical Product Innovation Project", China Mechanical Engineering, 2012, (01):1-6.

[5] “TMS320DM8127 and TMS320DM814x DaVinci™ Digital Media Processor" Technical Reference Manual, 2016, Texas Instruments

[6] Zhao Jiaxiang etc. “DSP system design and BIOS programming and application examples". Mechanical industry press. 2007

[7] “SYS/BIOS (TI-RTOS Kernel) v6.50 User's Guide" www.ti.com. 2017-10

[8] “Creating CCS Project for SysLink samples" www.ti.com. 2015

[9] “TMS320C674x DSP CPU and Instruction Set" Reference Guide, 2010(07) Texas Instruments

[10] “AM335x ARM CortexTM-A8 Microprocessors" TI. www.ti.com. 2015

[11] Li Jiajian. “Research and Implementation of Embeded CNC system based on OMAPL138 and FPGA". Guangdong University of Technology. 2015

[12] Spiros G. Papaioannou. “Interpolation algorithms for numerical control" Computers in Industry. 1979

[13] Xavier Beudaert, Sylvain Lavernhe,Christophe Tournier. “5-axis local corner rounding of linear tool path discontinuities" International Journal of Machine Tools and Manufacture. 2013