Received March 12, 2019, accepted April 3, 2019, date of publication April 9, 2019, date of current version April 19, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2909774

Architecture Modelling and Task Scheduling of an Integrated Parallel CNC System in Docker Containers Based on Colored Petri Nets

HONGYU JIN[1], YANG WANG[1], QIAN WANG[1], JIANKANG LIU[1], SHUHUA WANG[2], JUN ZHANG[2], SHANGHUA HAO[2], AND HONGYA FU[1]

[1]School of Mechatronics Engineering, Harbin Institute of Technology, Harbin 150001, China
[2]Foxconn Industrial Internet Co., Ltd., Shenzhen 518110, China

Corresponding author: Hongya Fu (hongyafu@hit.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 51805116, and in part by the Project of Open Intelligent Central Control System funded by Foxconn Industrial Internet Co., Ltd.

ABSTRACT With the diversification of product types and the development of complex CNC system functions, reconfigurable, and extensible CNC system must be developed to meet the needs of industrial production in the new era. To optimize the original architecture between the CNC system and the machine tool, and to improve the integration level of the industrial control system, an integrated parallel CNC system which could control multiple CNC machine tools based on high performance multi-core processor is proposed in this paper. Based on the virtualize container technology, the functional modules of the CNC system, such as the interpreter module and interpolation module, operate separately on each Docker containers. The system scheduling module uniformly schedules above functional modules and caches the data generated by each module to maximize the processing efficiency and resource utilization. For the modeling of system architecture, the colored Petri nets are used to establish the models of CNC system and task scheduling. Two strategies of task scheduling which based on task dependencies and data flow are presented to determine the priority of system function threads. According to the developed test platform, the real time of the virtual system environment, real-time system environment, and the communication module of integrated CNC are analyzed. Furthermore, test results show that the proposed integrated parallel CNC system has the ability of centralized control of two machine tools synchronously.

INDEX TERMS Integrated parallel CNC system, docker containers, colored Petri nets, architecture modelling, task scheduling.

I. INTRODUCTION

People's high requirements on processing efficiency, product quality make traditional closed manufacturing structure and management system difficult to cope with changing customer and product requirements. With the rapid development of electronic information industry, CNC machine tools which integrated with the advanced manufacturing system appeared [1]. In recent years, the rise of new technologies such as the Internet of things and cloud computing has brought modern industrial production to a new level. With the concept of cloud manufacturing proposed, the entire manufacturing system is moved to the cloud, and the integrated management of all CNC machine tools and production lines in the factory is an inevitable trend. CNC system of machine tools is the basic core device in the whole industrial manufacturing system [2]. Therefore, it is of great significance to study integrated CNC manufacturing system including its architecture modeling and integration methods.

Traditional CNC system adopts closed structure, which has poor portability, scalability and expansibility. The defect is that the equipment maintenance cost is high, and cannot meet the needs of product personalization [3]. Open CNC structure is a breakthrough of traditional CNC structure, which enables CNC system manufacturers to establish a wide range of cooperation on the standard platform. The open CNC system can be customized according to the needs of users, which greatly shortens the development cycle, and the

The associate editor coordinating the review of this manuscript and approving it for publication was Omar Khadeer Hussain.

open architecture also makes the operation and maintenance of the system more convenient. Wang *et al.* [4] presented an open CNC system which has an instruction format with double NURBS curves and suitable for 5-axis coordinated real-time interpolation. Based on an open controller and the standard of STEP-NC, Hu *et al.* [5] presented a feedrate optimization strategy in terms of 5-axis machine tool that can optimize machining parameters. To realize high speed and high precision real-time motion control of machine tool, the processor of CNC system needs to complete a lot of calculation in a short time. In addition, numerical control functions become more complex, and many additional modules need to be executed. With the development of hardware technology, the emergence of multi-core processor has greatly improved the performance of the processor and the PC based CNC system. Zhao *et al.* [6] proposed a non-uniform rational B-spline (NURBS) interpolator and motion controller which adopts the embedded multiprocessor. Tavares *et al.* [7] proposed a methodology for embedded hard real-time software synthesis with multiple operational modes based on a multiprocessor. Wang *et al.* [8] proposed a dynamic scheduling scheme which aims at scheduling real-time tasks on multiprocessor system-on-chip platforms. Davis and Burns [9] introduced a survey of hard real-time scheduling for multiprocessor systems. Above references studied the way of processing tasks in multi-core processor, but few of them combined the multi-core processor with the calling method of CNC system function module to build a parallel CNC system architecture. The utilization rate of multi-core processors in performing NC machining tasks need to be improved.

Since multi-core architecture has improved processor computing power and parallel processing capacity, multiple real-time applications can be integrated into a single multi-core processor platform, which conserves hardware resources and improves the overall system performance. With the increasing of integration and complexity, the design and optimization of multi-core processor based control system has become a more challenging work [10]. Traditional CNC system is a typical serial execution system. To make the control system run efficiently, it is necessary to make it run in parallel. Furthermore, in order to make full use of the computing resources of multi-core processors, a reasonable strategy should be developed to coordinate the task scheduling in CNC system. Mohammad and Entezari-Maleki [11] proposed a method for evaluation the grid service reliability based on the analysis of the task scheduling model. Zhang *et al.* [12] investigated a scheduling model for optimal production sequencing in a flexible assembly system. Jung *et al.* [13] analyzed cyclic scheduling problems of a timed Petri net to minimize the cycle time for automated manufacturing systems. Barreto *et al.* [14] presented two general approachesïijŁruntime and pre-runtime schedulingïijĽfor scheduling tasks in real-time systems. Ni *et al.* [15] proposed a resource allocation strategy for task scheduling based on priced timed Petri nets, by which the user can choose the satisfying resources autonomously from a group of preallocated resources. Baruah [16] proposed a new fixed-priority algorithm for scheduling systems of periodic tasks upon identical multiprocessors. Goossens *et al.* [17] presented a simulation interval for any schedule generated by a deterministic and memoryless scheduler for identical multiprocessor platforms in order to reduce the general asynchronous and arbitrary deadlines problem. Baek *et al.* [18] proposed a new preemptive fixed-priority scheduling algorithm to reduce the potential information leakage. For a set of periodic real-time tasks running on a single processor, Begam *et al.* [19] presented a task scheduling method which called Preference-Oriented Fixed-Priority (POFP). Runtime techniques based on slack management with dummy and wrapper tasks are also investigated with the objective of further improving tasks' execution preferences. Above references can improve the operation efficiency of the system by parallel task scheduling. However, CNC system has many functional modules and the tasks compete with the data in the cache area, so that frequent task blocking affects the real-time performance of the system. Therefore, the real-time performance of each functional module executed by the CNC system needs to be analyzed. In addition, the integrated CNC system using a high-performance computer to control multiple machine tools is rarely studied.

In view of above shortcomings, this paper plans to establish the architecture of the integrated CNC system first. To improve the execution efficiency, the system module is divided into several sub-modules to execute in parallel in each virtual containers of multiprocessor. Furthermore, to improve the utilization rate of resources and the efficiency of operation, this paper makes a task scheduling policy to schedule tasks reasonably. Colored Petri Net (CPN) is used to model the whole integrated CNC system from top to bottom, including modeling and analyzing the inter-task dependency and scheduling strategy. Finally, a numerical control container cluster was built based on container management technology to test the real-time performance and centralized control function of the presented integrated CNC system.

II. ARCHITECTURE MODELLING OF INTEGRATED CNC SYSTEM

A. ARCHITECTURE DESIGN OF INTEGRATED PARALLEL CNC SYSTEM

The program structure of most open source CNC systems is serial, such as GRBL and LinuxCNC. Although the basic control function module is included, the boundary between the function modules is blurred, which is not conducive to the parallel execution of the program. If the whole CNC system is put into the container, the performance of the system cannot be improved. Because the CNC system is multi-threaded internal structure, each thread to achieve specific functions. Assuming that CNC system runs on the single-core processor platform in the form of single-process multi-thread, then the system only runs on a single thread at any time, and other thread modules need to wait for resource scheduling. Other thread modules can only be scheduled to run when the

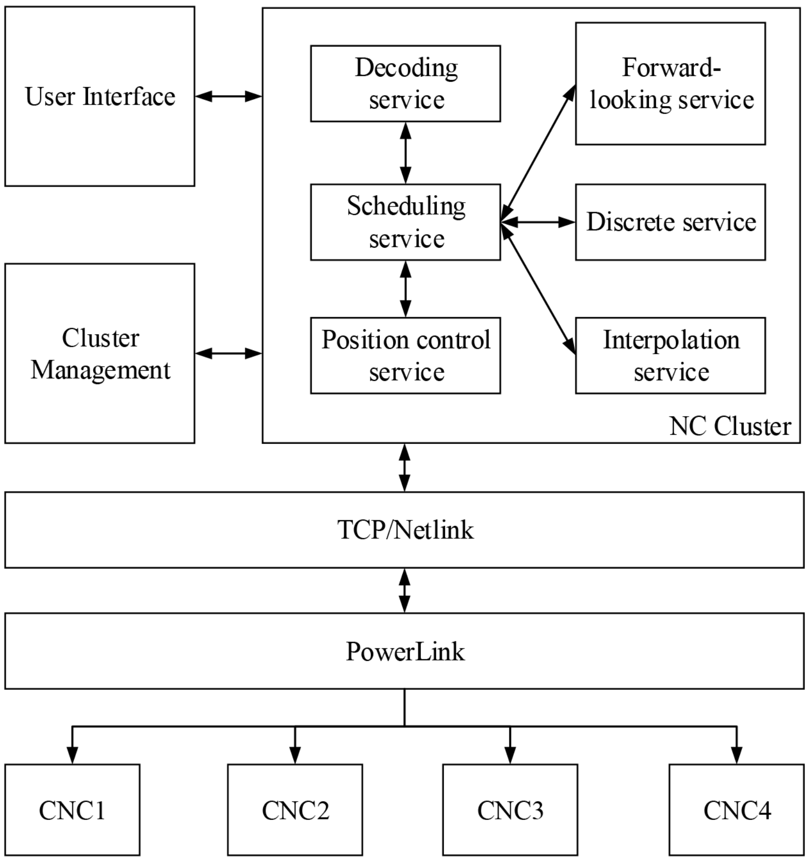


FIGURE 1. **Architecture of the integrated CNC system.**

running thread is dormant or suspended. If the CNC system runs on a multi-core processor platform, the maximum number of threads corresponding to the number of cores can be run at any time, which improves the performance of the CNC system. Therefore, increasing the number of processor cores can improve the performance of the system.

From the perspective of container technology, Docker container is a lightweight virtualization solution in multiprocessor. The idea is to ensure that a container only runs simple processes and provides a service. For most applications, especially for such a complex CNC system, it is necessary to use multiple containers to provide different services. Containers communicate with each other to form a container cluster to achieve specific functions. Based on the above explanation, the core functions of the CNC system are divided into several functional modules. Each module runs independently on a single container. For the integrated CNC system, there are a lot of modules providing the same function, and the number of containers in the whole cluster can reach hundreds. To achieve the management and maintenance of the system, such as expansion and resource management, manual operation for a single container is more difficult. So there must be a strong container management technology for unified management and scheduling. At present, Docker container management technology is mostly applied by Kubernetes, which integrates many advanced design concepts and extensive practical experience, and can easily manage thousands of containers. This paper is based on Kubernetes to manage the container cluster of integrated CNC system.

Service is the core concept of Kubernetes. The creation of a service can provide a unified entry address for a group of containers with the same function, and load balance the request and distribute it to each container application. In the integrated CNC system, the container modules that provide the same function of the numerical control system are encapsulated into a service and unified access interface is provided externally. For example, multiple interpreter containers are encapsulated into an interpreter service. According to the division of CNC system functions, the integrated CNC system can provide interpreter service, forward-looking service, discrete service, interpolation service, position control service and scheduling service. Among them, the scheduling service accepts the request of user interface, and position control service transmits location data to TCP communication module. Six services constitute the numerical control cluster, and the cluster management tool Kubernetes is used for unified management and scheduling.

The overall software architecture of the integrated CNC system is shown in Fig. 1. The topmost application consists of three parts: user interface, cluster management and numerical control cluster. The user interface can run on remote hosts or local hosts and connect the scheduling service of numerical control cluster through TCP/IP mechanism. Cluster management and numerical control cluster, as the master and slave nodes of the distributed system, can be deployed on one or more servers through the internal mechanism of Kubernetes. The lowest layer is the Powerlink real-time Ethernet communication layer, in which the Powerlink driver module runs in the operating system kernel and periodically sends position data and feedback status data to various CNC machine tool servo systems. Powerlink and CNC cluster are located in the kernel layer and application layer respectively, connected by TCP communication module. The position control service module of CNC cluster sends formatted data to TCP communication module through TCP/IP. TCP communication module uses Netlink communication mechanism between application layer and kernel layer to transfer data to Powerlink driver module.

Communication module plays a bridge role in the integrated CNC system, which is connecting the upper control system and the lower network protocol stack. After the user is connected to the numerical control service, the numerical control cluster automatically builds a numerical control processing data channel from interpreter to position control for the user, and the data is finally transferred to the position control module, which transmits control commands to the communication module and collects status and error information to the upper layer. Because CNC system runs in non-real-time virtual environment, the underlying network protocol runs in the host real-time environment, which requires the coordination of the middle layer mechanism. In addition, the Powerlink module connects all the servo drivers through a bus, and the data is collected and sent uniformly. So the centralization and separation of data in the middle layer are required. The software architecture of the communication module is shown in Fig. 2.

To connect real-time and non-real-time environment, TCP client process and the server process are executed in the virtual environment and hosting environment respectively. When multiple clients connected to the server, the server with every client to send and receive data cycle. In each client process, the client process through Shared memory and the position of the NC system control module for data interaction. Shared memory and TCP client in threads in the same period in the process of implementation, the Shared memory thread through Shared memory access to CNC system data in the ring buffer. The TCP client thread then sends data to the server periodically to consume the buffer data. In the server-side process, the number of ring buffers corresponding to the CNC system is maintained, which is used to store the data of each numerical control system received. On the other side of the buffer, the communication between Netlink inter-process communication mechanism and Powerlink module consumes the buffer data. TCP services and Netlink mechanisms are also implemented in server-side processes as two threads of

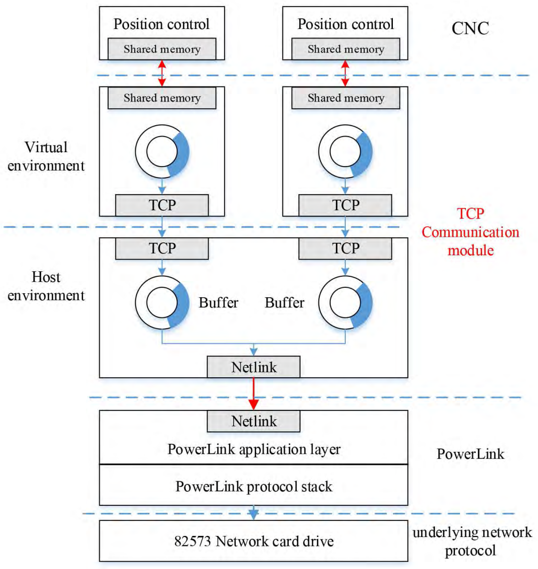


FIGURE 2. Software architechture of the communication module.

the same cycle. To ensure the performance of CNC system, the theoretical process of the client and server in the process of thread cycle should be equal, so as to guarantee the production and marketing of upstream and downstream data synchronization. However, the integrated CNC system is running in the virtualization of non-real-time environment, its thread to guarantee the real-time virtual environment of thread cycle jitter is greater than the threads in a hosting environment cycle jitter. Buffering mechanism can alleviate the difference between thread cycles and non-real-time environments.

Container technology obtains the virtualized environment of container isolation by virtualizing various resources of the host machine, one of which is network isolation. In Docker container, the network is implemented through a network library and various network forms are implemented with built-in drivers. There are mainly five built-in network drivers: brige driver, host driver, overlay driver, remote driver and null driver. The container uses the brige driver by default and connects to the Docker bridge by default at startup. In the host mode, the container can directly communicate with the outside world by using the IP of the host machine, and there is no problem of address conversion. Complexity of communication is greatly simplified and suitable for the integration of CNC efficient communication requirements. In the integrated CNC system, the virtual environment run by each numerical control system and the host environment run by the underlying network realize data interaction through TCP protocol. The host server-side program and the client program of each numerical control system have independent IP addresses. The biggest problem of TCP communication is the jitter of communication time. But the effect of jitter can be mitigated by using ring buffer. When the buffer data periodically sends data, the state feedback and receive the server. To connect the numerical control system with the underlying network, and make the data interaction between modules more efficient, each module adopts the unified IoState data structure for communication. It is important to note that the Shared memory mechanism is used to communicate in the Shared memory thread. The data interaction between the position control module and the communication module is realized by applying an IoState structure-sized memory block. In order for both modules to use Shared memory, mutexes are required to control that only one user at a time can manipulate Shared memory. On the basis of ensuring the same running cycle of the two modules, the data is accessed synchronously in the process.

B. COLORED PETRI NETS BASED CNC SYSTEM MODELLING

Petri net can describe the dynamic process of discrete events, and accurately describe the sequence, concurrency and conflict of events. Meanwhile, Petri net is based on strict mathematical theory, and many static and dynamic properties can be proved by mathematical methods.

A triad $N = (S, T; F)$ is called a network if it satisfies the following conditions.

$$\begin{cases} S \cup T \neq \emptyset \\ S \cap T = \emptyset \\ F \subseteq (S \times T) \cup (T \times S) \\ dom(F) \cup cod(F) = S \cup T \\ dom(F) = \{x \in S \cup T \,|\exists y \in S \cup T : (x, y) \in F\} \\ cod(F) = \{x \in S \cup T \,|\exists y \in S \cup T : (y, x) \in F\} \end{cases} \tag{1}$$

In Eq. 1, the element in S is called position, the element in T is called transition, and F is the flow relationship of network N. When Petri net is used to establish a real system model, the net describes the structure of the system and the token reflects the state of the system. M represents the token in each position in a certain state, called an identity of network N. A tetrad is called identity network. For the identity network, there are the following transition rules.

a) For the transition $t \in T$, t has the right to occur in the identification M if Eq.2 is satisfied, denoted as M $[> t$.

$$\forall s \in S : s \in {}^{\bullet}t \rightarrow M(s) \geq 1 \tag{2}$$

b) If M $[> t$ occurs, a new identity M' is obtained.

$$M'(s) = \begin{cases} M(s) - 1, when, s \in {}^{\bullet}t - t^{\bullet} \\ M(s) + 1, when, s \in t^{\bullet} - {}^{\bullet}t \\ M(s), other \end{cases} \tag{3}$$

$$\begin{cases} {}^{\bullet}x = \{y \,| y \in S \cup T \wedge (y, x) \in F\} \\ x^{\bullet} = \{y \,| y \in S \cup T \wedge (x, y) \in F\} \end{cases} \tag{4}$$

where ${}^{\bullet}x$ is the input set of x, and $x^{\bullet}$ is the output set of x.

The initial identifier of network system M_0 describes the initial state of the system. In the initial state, a transition occurs to obtain the new identity M_1. Under M_1, any transition takes place and a new identity M_2 is obtained. The transition cycle occurs and the identification changes make Petri net system run.

Colored Petri net (CPN) is a discrete event modeling language combining Petri net and Standard ML. Petri nets provide modeling with graphical annotations and basic primitives for concurrency, interactivity, and synchronization. Standard ML provides basic primitive descriptions for data type definitions, data manipulation descriptions, and the creation of compact parameterized models. CPN not only has the advantages of traditional Petri net graphical interface, semantic specification and universality, but also has strong description ability and simulation ability, which can simplify the modeling of the system. In addition, CPN adds the concept of time to capture how long it takes the system to process events. This means that CPN can be used to describe and verify system logic and performance, study system performance metrics, and also for real-time system modeling and validation.

The characteristics and functions of CPN are very suitable for modeling of CNC system. Time characteristics of CPN can be used to measure the performance of various tasks of CNC system, such as recording task execution time, waiting for resource time, etc. CPN can model CNC system at multiple levels of fine-grained and coarse-grained. Fine-grained can be the modeling at the program level, and then the modeling at the program level can be abstracted into modules to further abstract the system through the combination of modules. Coarse-grained can be the modeling at the module level or even the system level. Through these multiple layers, the internal structure and the overall structure of the system can be completely modeled.

The modules in the integrated CNC system based on Docker containers are independent each other. Communication mechanism is used to interact between containers, and the scheduling module calls each numerical control service module in turn for data processing. The functional module and scheduling module are considered as a whole, which is similar to the independent operation of a single CNC system, and the isolation of multiple CNC systems. To simplify the modeling process of integrated CNC system, the performance of the whole integrated numerical control system can be studied by establishing Petri net model of a single numerical control system. For the modeling of a single numerical control system, the hierarchical characteristics of CPN can be used to simplify the modeling process. In addition, some details of data processing and communication between modules need to be neglected in the modeling process at various levels.

Firstly, Petri net modeling is carried out for the top-level model of CNC system. Using the approach of building programs and the top-down design method, a set of hierarchical related CPN models with different functional modules are constructed. The basic idea of hierarchical model is to associate a substitution node with a module. Specifically, a transition is used to replace a module. When a module is associated with a transition, the module is called a submodule. In CPN modeling tools, transitions are identified by

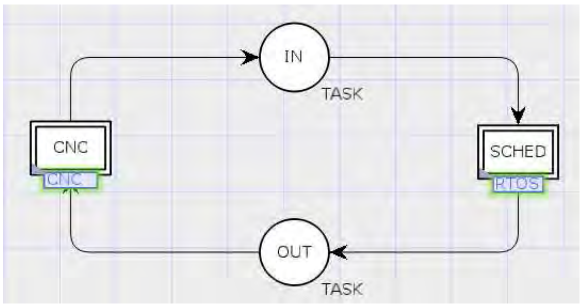


FIGURE 3. The top-level Petri net model of CNC system.

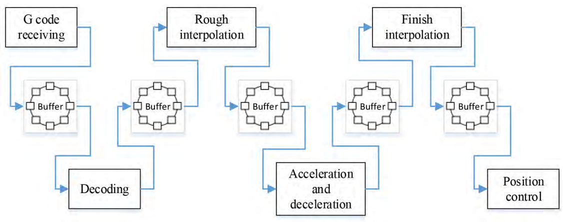


FIGURE 4. Functional modules of CNC system.

attaching labels. The tag contains the name of the submodule associated with the substitution nodes. In this paper, CNC system is divided into two modules: numerical control module (CNC) and scheduling module (SCHED). Figure 3 shows the top-level Petri net model of CNC system. Numerical control module and scheduling module are both vicissitudes. Two substitution nodes additional labels CNC and SCHED indicate that the names of these two submodules are CNC and SCHED respectively. The CNC module and scheduling module interact through the IN library and OUT library. The type of library is TASK, and the token in the library is TASK color set.

There is no clear boundary between the functional modules of traditional numerical control system, and there is data dependence between modules. The numerical control system is decomposed into the assembly line model with multiple numerical control tasks through the functional division of the numerical control system, and the functional modules of the numerical control system are decoupled. Each task is connected through a data cache. The cache upstream task produces data, and the downstream task consumes data. Upstream and downstream tasks and caches constitute the producer-consumer system. As shown in Fig. 4, the numerical control system is divided into six tasks: G code receiving task, interpreter task, discrete task, forward acceleration and deceleration task, interpolation task and position control task. Tasks are connected through a ring buffer, and upstream and downstream tasks need mutually exclusive access to the buffer. Each buffer specifies the format of the data.

The assembly line model of numerical control system is modeled. Each numerical control task is implemented as a submodule, and each task module is represented by a substitution node. Petri net model of CNC system is shown

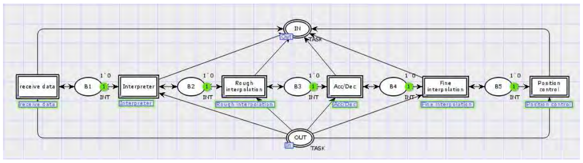


FIGURE 5. Petri net model of CNC system.

in Fig. 5. Receive Data, Interpreter, Rough Interpolation, Acc/Dec, Fine Interpolation and Position represent the submodules of receiving G code task, interpreter task, discrete task, acceleration and deceleration task, interpolation task and position control task respectively. B1, B2, B3, B4 and B5 are data buffer libraries corresponding to corresponding tasks. The type of library is INT, which represents the data buffer stock of this buffer. Each data cache library has only one token, and the buffer can be used by the upstream and downstream tasks mutually exclusive. The task token generated by each task transition enters the scheduling system through the IN library for scheduling operation, and then re-enters the corresponding task transition through the OUT library after the operation, indicating that the data processing is completed once. IN and OUT libraries represent the communication process between tasks and scheduling modules in the numerical control module.

The modeling process of each numerical control task submodule is as follows. In the modeling process, the details of data processing and the interaction between numerical control task and buffer module are ignored, so that the program structure of each numerical control task is similar. CNC tasks are mainly divided into three types: the upstream task receiving G code task only generates data, the downstream task position control task only consumes data, and the intermediate task needs to consume data and produce data. First, model the task receiving G code, as shown in Fig. 6. After the task is initialized, listen for the user request and connect the user to receive the G code. When you receive code, you need to check whether the downstream data buffer is free. Receive if idle, otherwise wait for buffer data to be processed by downstream tasks. This loop receives until all G code is received and the task is completed and disconnected from the user. In the Petri net model for receiving G code tasks, the token in the waitdata library is used to indicate that the task is waiting for data. Tokens in the record library and the datarec library are used to record state data for the task. Init transition refers to the task initialization process, and a specified amount of delay can be attached through "@". Start transition refers to the start of data processing, complete transition refers to the completion of data processing. Meanwhile, the data of this task instance is output to the record library and datarec library.

Intermediate numerical control tasks include interpreter task, discrete task, acceleration and deceleration task, interpolation task. The program structure is the same, the difference is the data processing part. Interpreter task to obtain a complete G code, code preprocessing, filtering whitespace, comments and other meaningless characters. The conditional

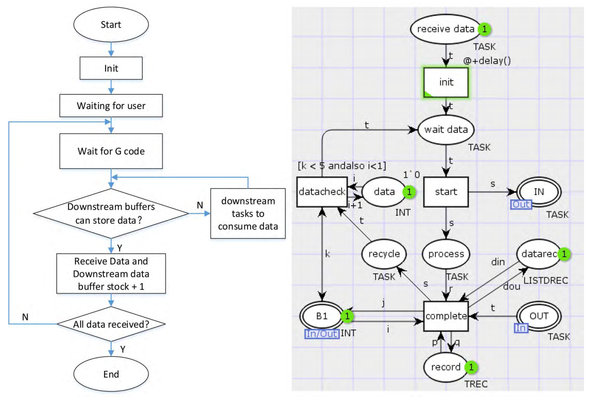


FIGURE 6. The model of G code receiving task.

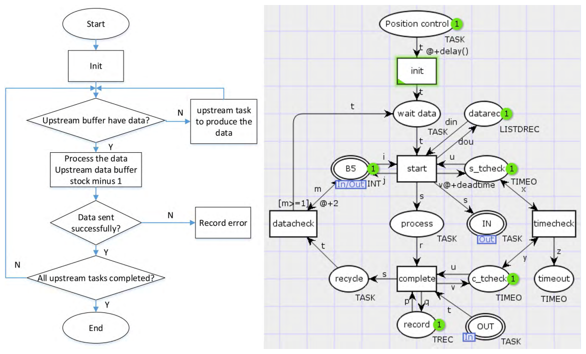
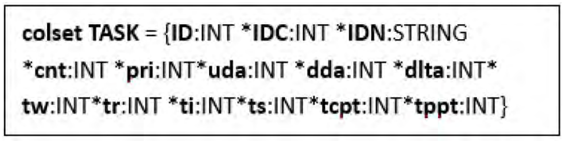
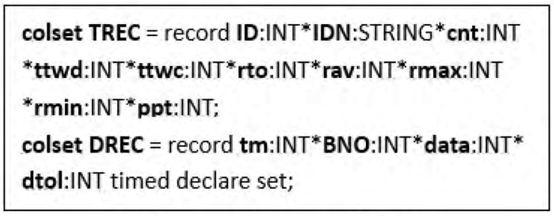


FIGURE 7. The model of position control.


colset TASK = {**ID**:INT ***IDC**:INT ***IDN**:STRING ***cnt**:INT ***pri**:INT***uda**:INT ***dda**:INT ***dlta**:INT* **tw**:INT***tr**:INT ***ti**:INT***ts**:INT***tcpt**:INT***tppt**:INT}

(a)


colset TREC = record **ID**:INT***IDN**:STRING***cnt**:INT ***ttwd**:INT***ttwc**:INT***rto**:INT***rav**:INT***rmax**:INT ***rmin**:INT***ppt**:INT;
colset DREC = record **tm**:INT***BNO**:INT***data**:INT* **dtol**:INT timed declare set;

(b)

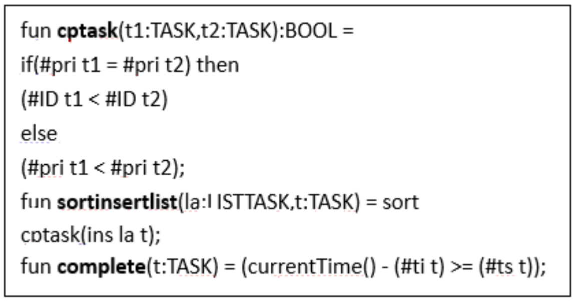
FIGURE 8. Color set definition of CNC system.

branching statement is used to judge the meaning of each program block one by one and the corresponding information is written into the numerical control program block information structure. For tool movement, the numerical control system stores straight line and arc parameters into the downstream cache. For auxiliary function instructions, the relevant flags are set. The interpolation task interpolates the line segment or arc segment of the upstream cache area. The system connects the densification points with the line segment and successively stores the line segment information into the downstream buffer. The forward acceleration and deceleration task deals with the abrupt change of velocity at the junction of the adjacent straight segment, so that the movement of the cutter at the junction of the straight segment can meet the acceleration and deceleration characteristics of the servo system. Combined with the rated speed of the straight line, acceleration and deceleration planning is carried out, and trapezoidal acceleration and deceleration planning strategy is adopted in the system. Finally, acceleration and deceleration curve information is stored in the downstream buffer. According to acceleration and deceleration curves, interpolation tasks divide each straight line segment into micro straight line segments. The increment of each axis position is calculated and the pulse amount is put into the downstream buffer.

Position control task is the key task of numerical control system, and real-time performance determines the performance of numerical control system. Its function is to periodically send the number of axis pulses obtained by interpolation to the real-time Ethernet Powerlink module. The modeling of position control tasks is shown in Fig. 7. Tasks consume only upstream data, and if the upstream data buffer is empty, wait for the upstream task to produce data. The position control task requires high real-time performance and detects whether the data is sent successfully in the modeling process. In Petri net model, datacheck transitions detect whether there is data in the upstream data buffer, and wait for the upstream task to produce data if it is empty. Timecheck transition detects the time difference between the start of the task and the completion of data transmission. If the timeout occurs, the timeout information is recorded through the timeout library.

To describe real-time tasks and upstream and downstream data cache information in the model, a set of related color set tasks is defined to record the status information of the task at different times (as shown in Fig. 8a). In addition, TREC color set (Fig. 8b) is used to record some information of the completed task, and DREC color set is used to record the instantaneous data amount and accumulated data amount of each data buffer.

Each parameter in the TREC color set is processed by complete transition, and its parameter corresponds to the parameter in the TASK color set. The ID represents the task number. The IDN represents the task name. CNT represents the cumulative number of instances of a task. TTWD represents the accumulated time for waiting for data resources, and TTWC represents the accumulated time for waiting for processors. RTO represents the total response time. RAV represents the average response time. rmin and rmax represent the minimum response time and the maximum response time respectively. PPT represents the cumulative number of preemptions. All parameters of DREC color set are processed by ready transition and complete transition. Tm represents the current time and BNO represents the data buffer number. Data represents the instantaneous data volume of buffer and DTOL represents the accumulated data volume of buffer. In addition, CPN defines functions for token processing. In the scheduling



```
fun cptask(t1:TASK,t2:TASK):BOOL =
if(#pri t1 = #pri t2) then
(#ID t1 < #ID t2)
else
(#pri t1 < #pri t2);
fun sortinsertlist(la:LISTTASK,t:TASK) = sort
cptask(ins la t);
fun complete(t:TASK) = (currentTime() - (#ti t) >= (#ts t));
```

FIGURE 9. Definition of functions in scheduling module.

module, three functions are defined as shown in figure 9. The function of "cptask" compares task priorities and returns BOOL variables. The function of "soninsertlist" inserts a new or preempted task into a task ready queue by priority, returning the task ready queue after the inserted task. The function of "complete" determines whether the task is complete and returns a BOOL variable.

III. TASK SCHEDULING OF INTEGRATED CNC SYSTEM

A. ANALYSIS AND DESIGN OF TASK SCHEDULING SCHEME

The scheduling module is the core module of the integrated CNC system. The main functions are to receive user requests, cache the numerical control task data and implement the container scheduling strategy. The scheduling service listens to the request sent by the user and forwards the traffic to a scheduling module. If no idle scheduling module receives the request, the user needs to wait. When the user connects to the scheduling module, the scheduling service binds the user to the scheduling module, which is equivalent to the user connecting to a CNC system. Subsequent data from the user is forwarded to the module until the user disconnects. After establishing the connection with the user, perform the corresponding operation according to the command parameters sent by the user. If it is an auxiliary function command, such as spindle command, coolant command, etc., directly transfer the command to the execution system. If it is a processing command, the interpreter module is called to start the G code interpretation. Then the discrete module, forward acceleration and deceleration module, interpolation module, position control module and other numerical control modules were successively called for data processing. The scheduling module maintains a series of ring buffers internally for the numerical control system and stores the input and output data of each functional module in a specified format. For example, the interpreter module does not directly store input G code and output line segment information. The scheduling module sends the G code to the interpreter module, and the interpreter module parses the G code and sends the line segment information to the scheduling module to complete a data processing. Each functional module on the implementation runs in a separate docker container.

In addition to data caching, the scheduling module can also perform numerical control function module scheduling, namely container scheduling. The significance of scheduling lies in coordinating the data processing of each container, improving the real-time performance of CNC system and resource utilization of the cluster. The scheduling here is similar to traditional task scheduling. For traditional scheduling, the operating system manages the limited resources of the system. To control the number of resource users, process occupancy must be selected according to certain scheduling principles to make the system run orderly and improve system performance. In the cluster of CNC system, when the system resources are tight, the scheduling strategy can be developed to coordinate the operation of various containers to improve the system performance. This paper draws on the design idea of traditional task scheduling strategy to develop container scheduling strategy suitable for integrated CNC system.

Integrated CNC system can be divided into three types of tasks: non-periodic tasks, soft real-time tasks and hard real-time tasks. Among them, G code receiving task is aperiodic task, and position control task is hard real-time periodic task. In fact, time performance determines the performance of numerical control system, and the highest priority should be assigned to ensure the real-time performance of position control. Soft real-time periodic tasks include interpreter, discrete, forward acceleration and deceleration and interpolation tasks, which have upstream and downstream data dependence with position control tasks.

Two priority allocation schemes are compared and analyzed in this paper. One is to assign static priority according to the strength and weakness of each soft real-time task and position control task data dependency based on task dependency. Priorities are assigned and remain the same while the task is running. The other one is based on data flow, and dynamically prioritizes the allocation by analyzing the data volume of each data cache area. Tasks are initialized with an initial priority that is dynamically adjusted based on the amount of data in the cache.

For priority allocation strategies based on task dependency, interpolation task, forward task, discrete task and interpreter task have weaker and weaker data dependence with position control task. So the tasks assigned to them decrease in priority. That is, interpolation tasks are assigned higher priority and interpreter tasks are assigned lower priority. The task scheduling process is shown in Fig. 10. The data is traversed by task number from high-priority to low-priority tasks. If there is data in the upstream cache and data can be stored in the downstream cache, the task execution is scheduled. For the priority allocation strategy based on data flow, the task priority is determined according to the data volume difference between upstream and downstream data cache. The scheduling process is shown in Fig. 11, and the task is traversed by task number. If the upstream cache area of a task has no data or the downstream cache area cannot store data, the task will not be scheduled this time. If the upstream cache area has data and the downstream cache area can store data, the data volume difference between the upstream and

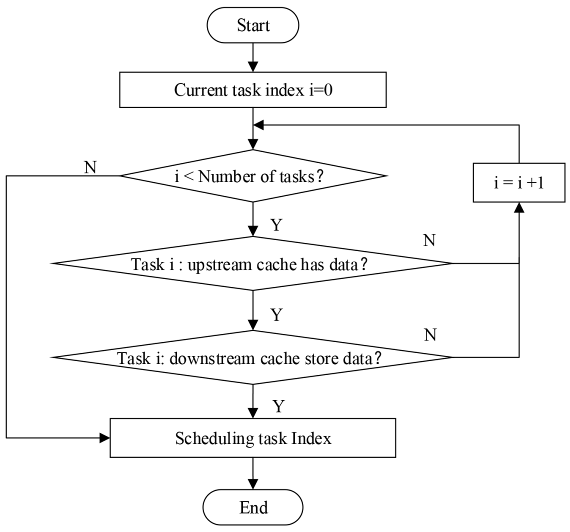
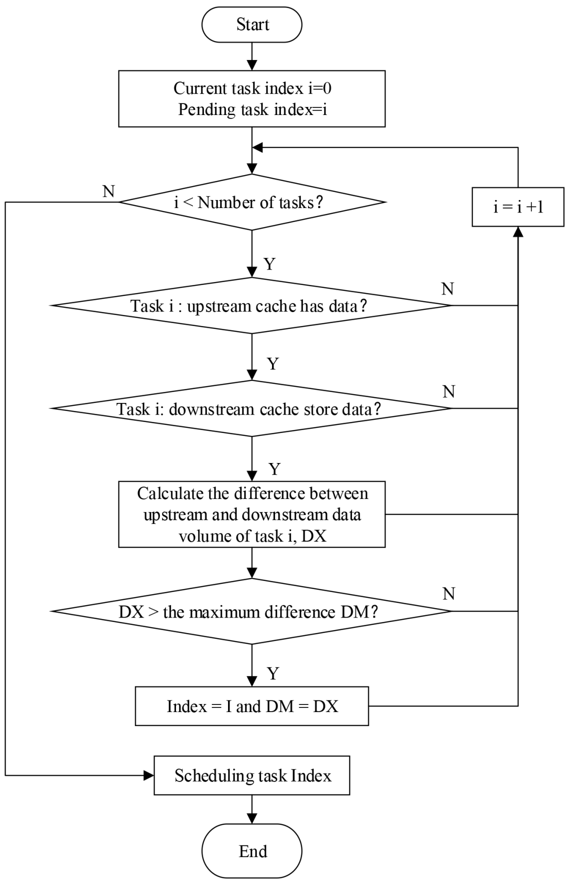


FIGURE 10. Task scheduling process based on task dependency.



Start
Current task index i=0
Pending task index=i
i < Number of tasks?
i = i +1
N
Y
Task i : upstream cache has data?
N
Y
Task i: downstream cache store data?
N
Y
Calculate the difference between upstream and downstream data volume of task i, DX
DX > the maximum difference DM?
N
Y
Index = I and DM = DX
Scheduling task Index
End

FIGURE 11. Task scheduling process based on data flow.

downstream cache area of this task is calculated and the maximum value of this value is recorded. After all tasks are traversed, the schedulable task with the largest data volume difference between upstream and downstream cache regions can be obtained and scheduled.

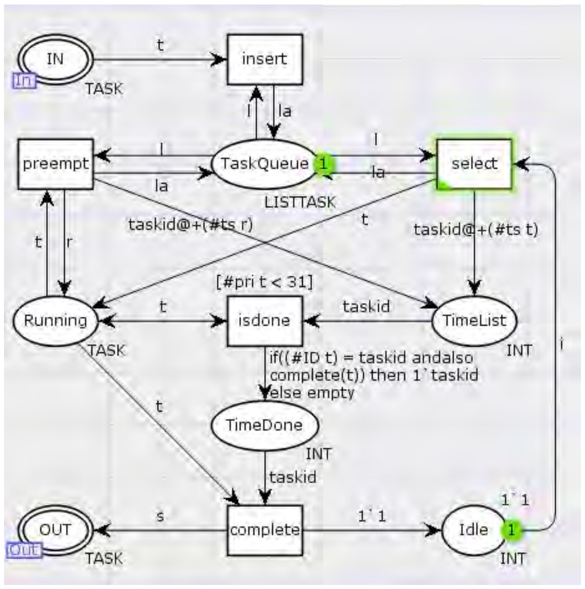


FIGURE 12. Petri net model of scheduling submodules.

B. PETRI NETS BASED TASK SCHEDULING OF INTEGRATED CNC

In the top-level Petri net model of CNC system, the scheduling module is implemented as a submodule by substitution node SCHED. Petri net model of scheduling submodules is shown in Fig. 12.

In Petri net model of scheduling module, one library and three transitions are more important. The TaskQueue library is of type LISTTASK, where the token represents the task-ready queue, which stores tasks according to task priority. The ready queue operated by the Insert transition, Preempt transition, and Select transition, and the Insert transition inserts a task into the ready queue after it reaches the IN library. Preempt transitions perform preemptive actions when a high-priority task is ready, while updating task parameters such as the first time a task enters the processor, the time it waits for the processor, the time it is preempted out of the processor, the remaining execution time, and the number of preempts. Select transition tasks complete and select the high-priority tasks in the task ready queue to run. The above two scheduling strategies can be implemented by the Preempt transition and Select transition description functions, so the Petri net model can be shared to model the two scheduling strategies.

The Running library is of type TASK, where the token indicates that a TASK is Running. The TimeList library is of type INT, where the token represents the execution time of the task. Preempt transitions and select transitions both take the highest priority task from the ready queue and output the task token to the Running library. At the same time, the task number is output to the TimeList library as a token by attaching delay information through "@". The delay information is the sum of the current time of the system and the remaining execution time of the task, which represents the time that the task still needs to execute. Preempt transitions could inserts

TABLE 1. Simulation parameters of six tasks.

Task	1	2	3
Task name	G code receiving	Interpreter	Discrete
Priority	3	3	3
Execution time	10 ms	10 ms	10 ms
Task cycle	-	5 ms	5 ms
Task	**4**	**5**	**6**
Task name	Forward-looking	Interpolation	Position control
Priority	3	3	1
Execution time	10 ms	10 ms	5 ms
Task cycle	5 ms	5 ms	2/10/15 ms

unfinished low-priority tasks into the ready queue for subsequent execution. The isDone transition determines whether a task has finished execution by comparing the current time with the delay time of the task, and then outputs the task number as a token to the TimeDone library. The Complete transition is used to record the completion time of the task, move the finished task from the Running library to the OUT library, and generate tokens representing the Idle resource into the Idle library. The Select transition is activated and the highest priority task is selected from the task ready queue to run.

Based on task scheduling strategies of task dependency and data flow, the process of data execution and consumption in the system is simulated. The position control task in CNC system is a hard real-time periodic task. The simulation takes the task cycle of position control as a variable and observes the data flow changes under different task cycles. Data dependent tasks include interpreter task, discrete task, forward-looking task and interpolation task. Task cycle of position control should be determined by analyzing changes in the buffer data. The parameter settings of simulation are shown in table 1.

In the simulation, the task cycles for position control are set to 2 ms, 10 ms, and 15 ms respectively. The data volume of each data buffer changes with time as shown in Fig. 13.

From Fig. 13, when the position control task cycle is 2 ms, the execution cycle of the data dependent task is greater than position control task, and the data consumption speed of the position control task is greater than the data generated by the data dependent task. The data volume of each task buffer fluctuates frequently around 0, indicating that the position control task often waits for upstream data. When the position control task cycle is 10 ms, the data dependent task execution cycle is equal to the position control task cycle. At this point, the speed at which the position control task consumes data is approximately equal to the speed at which the data-dependent task generates data. Comparative analysis shows that the data balance effect of scheduling strategy based on data flow is better. The buffer data volume fluctuates in a position far greater than zero, resulting in shorter running time of each task and higher efficiency of the system. Part of the data dependent task fluctuates frequently around the data volume 0, indicating that the task frequently waits for the upstream data and the system operation efficiency is low. When the position control task cycle is 15 ms, the data

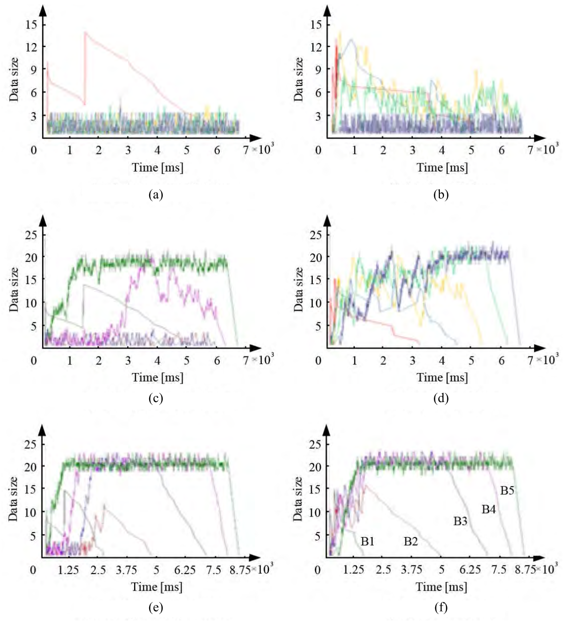


FIGURE 13. The data volume of each data buffer.

dependent task execution cycle is greater than the position control task cycle. Part of the data based on task dependent scheduling fluctuates a little in the early stage, but does not affect the position control task. It can be seen from the simulation that the task scheduling strategy based on data flow has an obvious data balance effect only when there is a small difference in the production and consumption speed between the position control task and the data-dependent task data. In general, the task scheduling strategy based on data flow has a better effect than the scheduling strategy based on task dependency.

IV. EXPERIMENTAL VALIDATION

A. REAL-TIME PERFORMANCE ANALYSIS OF INTEGRATED CNC

CNC machine tool has a system of strong real-time performance. Running on the general processor platform usually requires the system to provide high-precision real-time clock. Virtual environment runs in the application layer of the operating system. It has no access to the operating system kernel and can only run in non-real-time environment. The integrated CNC system proposed in this paper runs on the virtual Ubuntu operating system. In order to better study the integrated CNC system, the real-time performance in the virtual environment was first tested. The host machine runs the Debian operating system and extends the real-time kernel, and runs the OpenPowerlink protocol stack and the underlying network protocol with high real-time requirements. The virtual environment where CNC system is located and the host environment where Powerlink is located are connected by TCP communication module. The real-time performance of the communication module is also crucial, so the real-time performance of the TCP communication module needs

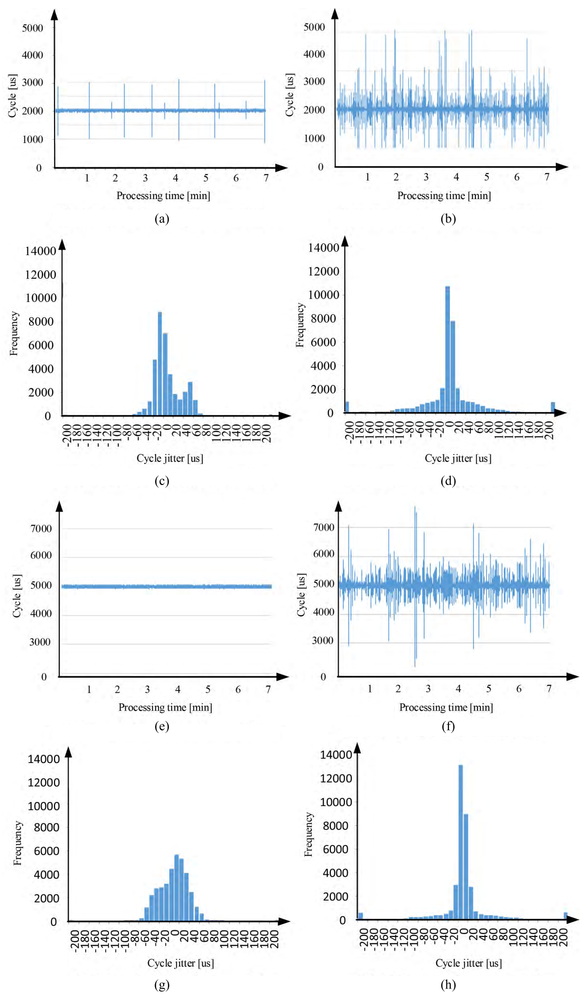


FIGURE 14. **The jitter of position control in real time and virtual environment.**

to be tested. The real-time performance evaluation index is the jitter size of the periodic task, which is defined as the difference between the maximum and minimum values in the actual cycle cycle. This paper tests the real-time performance of the virtual environment, Powerlink module and TCP communication module respectively.

(1) Real-time performance test of virtual environment

Position control is the last part of the operation of CNC system. Firstly, the real-time performance of position control in virtual environment is tested. At the same time, the CNC system with the same configuration is guaranteed to run on the host machine with real-time environment by means of comparative study. The operation condition is compared with that in the virtual environment. The test uses a common G code that includes arc and line interpolation. When the position control cycle is 2 ms and 5 ms respectively, the actual execution cycle of each position control cycle is counted. According to the statistical results, the jitter of position control cycle in real time environment and virtual environment is obtained, as shown in Fig. 14.

From the statistical results of position control period of 2 ms, the jitter is generally maintained within the range of $[-60, 60]$ μs in the real-time environment. CNC system as a real-time process running in the system, its priority is higher than the priority of ordinary processes. Occasionally large jitters occur because processes with higher priority are running. In virtual environments, the jitter changes mostly to [-120, 120] μs, but there are some changes over 200 μs. In the virtual environment, the CNC system only as a normal process, cannot take precedence over other processes to run. When other processes occupy resources, the CNC process must wait for the release of resources to run. So the position control cycle jitter is large. For the phase control period of 5 ms, the position control cycle jitter is maintained at [-60, 60] μs in the real-time environment. The CNC system operates very smoothly, and the jitter will not exceed the range. It can be seen that the system has sufficient resources to run the CNC system at 5 ms. In the virtual environment, the jitter range is still very large, the jitter frequency over 200 μs is still very high.

According to the above statistics and analysis, the real-time performance of the CNC system in the virtual environment is not as good as that in the real-time environment. But for the purpose of integrated control, CNC systems must be isolated from each other. From the point of view of current realizable scheme, it is the best choice to run CNC system in virtual environment. Furthermore, from the perspective of a small machining process, the average cycle of multiple cycle execution of position control fluctuates in a small range. Data buffering can be used to solve the problem of insufficient real-time performance of the system.

(2) Real-time performance test of Powerlink module

Powerlink module is divided into two parts: application layer and driver layer. This paper adopts OpenPowerlink software implementation scheme. The application layer and driver layer run at the user layer and kernel layer of the system respectively. OpenPowerlink needs to run on a real-time system. Therefore, it is necessary to install the Linux real-time kernel extension before adopting this scheme. In addition, the performance of Powerlink protocol stack implemented by software scheme is greatly affected by computer performance. Because Powerlink's behavior is triggered by timers, the system must provide a high-precision timer and a quick interrupt response. Therefore, the timer precision provided by the system often determines the real-time performance of Powerlink.

In order to test the real-time performance of Powerlink under the integrated CNC system, firstly, the virtual environment should be set up and two numerical control systems should be run on it. Run the communication module and Powerlink module in the host environment. The host machine was configured as a management node using Open-Configurator (a networking tool of Powerlink), and the six servo drivers corresponding to the two CNC systems were configured as control nodes. The nodes are connected by bus topology, the communication cycle is 10 ms, and the two numerical control systems run a common numerical control code respectively. The variation of the cycle with the

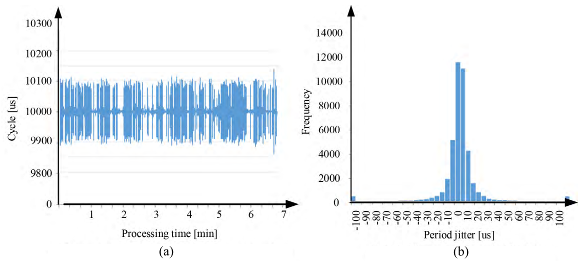


FIGURE 15. Cycle jitters on the Powerlink application layer.

processing time and the frequency of each jitter range are shown in Fig. 15. Cycle jitters on the OpenPowerlink application layer vary is [-100,100] μs. The jitter is relatively small compared with the position control cycle jitter in the virtual environment, so integrated CNC machining experiments can be carried out.

The real-time test of Powerlink underlying driver network was carried out. The underlying driver network controls the timing of data transmission, and the jitter time is microseconds or even nanoseconds. Therefore, bus analysis tool is needed to analyze the timing sequence of data frames accurately. In this paper, industrial Ethernet bus analyzer Allbustap is adopted, which can accurately record the TAP moment of data frame passing through. The timestamp resolution is up to 1 ns, and the fetching process does not change the data frame, nor does it cause additional jitter and delay. Placing the analyzer between the master station and the first slave station captures the exact time elapsed for each frame of data on the Powerlink bus. It can be further known that Powerlink isochronous phase adopts Preq/Pres mode in the integrated CNC system, and the communication data frame length between nodes is 80 bytes. When the NC code is finished, the captured data frame information can be analyzed. In order to test the real-time performance of the bus, the Cycle jitter of the management node sending data to each slave node is analyzed. The six servo nodes in the integrated CNC system are connected by bus, with node 1 nearest to the management node and node 6 farthest from the management node. Node 1, node 3 and node 6 were selected for jitter analysis respectively, and 2000 frame data received by the three nodes were analyzed respectively. The results are shown in Fig. 16. It can be seen that the node 1 jitter near the main station stays at [-6, 6] μs. For the node 6 farthest from the main station, the jitter range is large, but most of it stays between [-10, 10] μs. Node 3 is the intermediate node, and the jitter is between node 1 and node 6. To sum up, the Cycle jitter of each node is basically maintained between [-10, 10] μs. It can be used for machining experiments in the proposed system.

(3) Real-time performance test of TCP communication module

As an intermediate module, TCP communication module has the functions of communication connection and data buffering. It connects the virtual environment numerical control system to the host's Powerlink module. The data buffer includes two parts, one is the data buffer between the numerical control system and the communication module, the other is the data buffer between the communication module and the Powerlink module. TCP communication cycle is determined by the data sent by the client, and the data received by the server follows the change. Therefore, the real-time test of TCP client process can represent the test of the whole module. The test runs two CNC systems in a virtual environment, and the location instruction data is transmitted to the underlying Powerlink network through the TCP communication module. In order to fully understand the running performance of TCP communication module, the communication cycle is set as 2 ms, 5 ms and 10 ms, and 20,000 times of data sent in the three cases are statistically analyzed. The results are shown in Fig. 17.

From Fig. 17, the Cycle jitter of data transmitted by TCP communication module is relatively large, and the uncertainty of communication process is relatively high. Cycle jitter is basically greater than zero, indicating that sending and receiving data takes a lot of time. There are many factors that affect TCP communication, including processor performance, the number of TCP connections, network environment, and the amount of communication data. It is necessary to observe whether the integrated CNC system has obvious jitter in the process of running NC code. So the influence of communication instability on the machining process can be eliminated by reasonably determining the communication cycle.

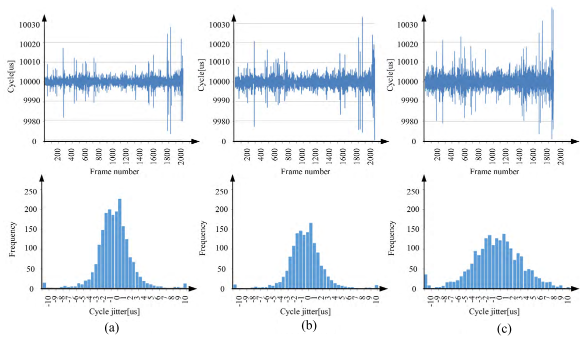


FIGURE 16. Jitter analysis of real-time test of Powerlink underlying driver network.

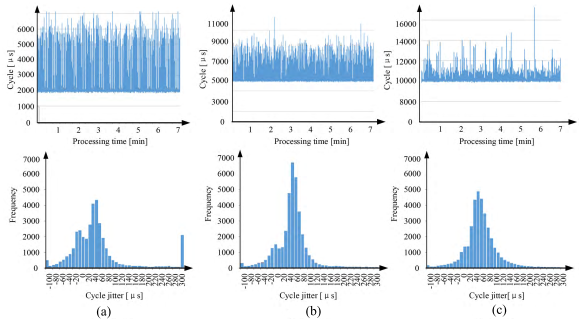


FIGURE 17. Real-time performance test of TCP communication module.

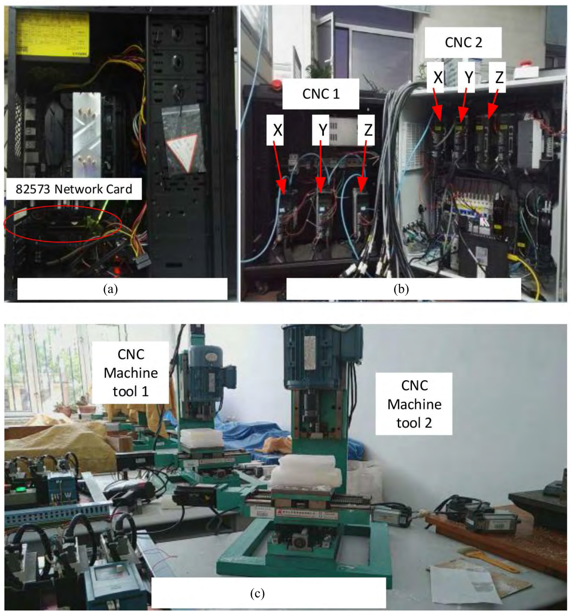


FIGURE 18. The hardware platform of the integrated CNC.

B. EXPERIMENTAL VALIDATION OF INTEGRATED CNC MACHINING

Through the real-time test of each module of the system, except for the small jitter of the application layer and driver layer of Powerlink module, the real-time performance of the CNC system and the client module in the virtual environment under the cycle of 2 ms and 5 ms was not good. When the period is 10 ms, the jitter of non-real-time CNC system and TCP communication module is relatively small. In order to ensure the successful completion of numerical control machining experiments, the system cycle is 10 ms. In the actual operation process, the cycle of each module of the system should be consistent, including the position control cycle of the CNC system, the transmission data cycle of TCP communication module and the communication cycle of Powerlink module. For TCP communication, the period setting should be slightly less than 10 ms because it takes about 100 μs to send data and receive status flags. Compensation is made for the time of sending and receiving data. Because there is an internal ring cache for data dynamic balance, the whole machining process can be smoothly carried out.

The hardware platform of the machining experiment of the integrated CNC system is shown in Fig. 18. Figure 18a shows a high-performance general-purpose multi-core processor (AMD I686, 12 cores, 2.2GHz, 8GB of memory, 2TB of hard disk) extending the 82573 network card on its PCI slot as the physical communication layer of OpenPowerlink. The host environment runs the Debian system and extends the real-time kernel, as well as the virtual environment, the OpenPowerlink application layer, and the TCP server process module. The virtual environment runs two Ubuntu systems, each running a CNC system and TCP client process module. Figure 18b shows two electrical control cabinets of the integrated CNC system, each equipped with three servo drivers supporting the Powerlink protocol. Figure 18c shows two CNC milling machines used in the machining experiment.

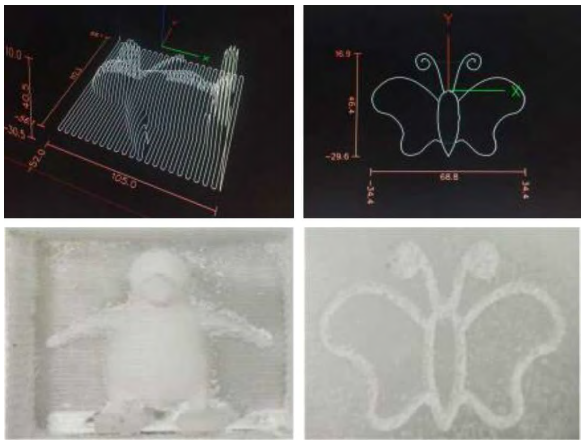


FIGURE 19. Tool path and machining result by integrated CNC.

After the system hardware is properly connected, wax molds of 15cm*15cm*5cm in length, width and height are installed in the two small milling machines for subsequent processing. The G code of processing trajectory of penguin and butterfly shapes is generated by CAM software, and the tool path and machining result are shown in Fig. 19.

The penguin and butterfly test-piece processed by two CNC milling machines are respectively controlled by the CNC system 1 and CNC system 2 in the integrated CNC system. It can be seen that the finished surface is consistent with the trajectory planning, which verifies the feasibility of the proposed integrated CNC system.

V. CONCLUSION

Based on virtualized container technology, an integrated parallel CNC system which could run multiple numerical control system in high performance multiprocessor is proposed. The overall architecture of the integrated CNC system is established, each functional module of the system runs independently in its own Docker container. Modelling of functional modules in proposed system is achieved by CPN, the data interaction between the modules of the system is realized through substitution nodes in Petri net. Two task scheduling strategies which based on task dependency and data flow are proposed. From the simulation, the task scheduling strategy based on data flow has positive effect on data balance. Real-time performance tests in terms of system virtual environment, Powerlink module and communication module are carried out. The test shows that the cycle jitter of the underlying Powerlink driver module is small and stays within 10 μs, but how to reduce the Cycle jitter of communication module needs further research. From the processing experiments, the proposed integrated CNC system can control two machine tools at the same time, which lays a theoretical and practical foundation for the centralized CNC system to control multiple machine tools in the future.

Future research plan: To reduce the data dependent execution cycle and to achieve a balanced data transfer between the position control task and the data dependent execution task. To guarantee that the multi-core platform can handle more tasks and function modules, we will focus on improving processor performance, memory, network card communication speed.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their careful, constructive and insightful comments to improve the quality of the present contribution.

REFERENCES

[1] Z. U. Rehman, O. K. Hussain, and F. K. Hussain, "Parallel cloud service selection and ranking based on QoS history," *Int. J. Parallel Program.*, vol. 42, no. 5, pp. 820–852, 2014.

[2] C.-S. Chen and T. S. Yin, "Intelligent computer-aided process planning of multi-axis CNC tapping machine," *IEEE Access*, vol. 5, pp. 2913–2920, 2017.

[3] N. K. Giang, R. Lea, and V. C. M. Leung, "Exogenous coordination for building fog-based cyber physical social computing and networking systems," *IEEE Access*, vol. 6, pp. 31740–31749, 2018.

[4] Y. Wang, X. Ma, L. Chen, and Z. Han, "Realization methodology of a 5-axis spline interpolator in an open CNC system," *Chin. J. Aeronaut.*, vol. 20, no. 4, pp. 362–369, 2007.

[5] P. Hu, Z. Y. Han, H. Y. Fu, and D. D. Han, "Architecture and implementation of closed-loop machining system based on open STEP-NC controller," *Int. J. Adv. Manuf. Technol.*, vol. 83, nos. 5–8, pp. 1361–1375, 2016.

[6] H. Zhao, L. Zhu, Z. Xiong, and H. Ding, "Development of FPGA based NURBS interpolator and motion controller with multiprocessor technique," *Chin. J. Mech. Eng.*, vol. 26, no. 5, pp. 940–947, 2013.

[7] E. Tavares *et al.*, "A time Petri net based approach for embedded hard real-time software synthesis with multiple operational modes," in *Proc. 18th Symp. Integr. Circuits Syst. Design*, 2005, pp. 98–103.

[8] C. Wang, X. J. Feng, X. Li, X. H. Zhou, and P. Chen, "Colored Petri net model with automatic parallelization on real-time multicore architectures," *J. Syst. Archit.*, vol. 60, no. 3, pp. 293–304, 2014.

[9] R. I. Davis and A. Burns, "Improved priority assignment for global fixed priority pre-emptive scheduling in multiprocessor real-time systems," *Real-Time Syst.*, vol. 47, no. 1, pp. 1–40, 2011.

[10] H. Fu, J. Liu, Z. Han, and Z. Shao, "A heuristic task periods selection algorithm for real-time control systems on a multi-core processor," *IEEE Access*, vol. 5, pp. 24819–24829, 2017.

[11] A. A. Mohammad and R. Entezari-Maleki, "Task scheduling modelling and reliability evaluation of grid services using coloured Petri nets," *Future Gener. Comput. Syst.*, vol. 26, no. 8, pp. 1141–1150, 2010.

[12] W. J. Zhang, T. Freiheit, and H. Yang, "Dynamic scheduling in flexible assembly system based on timed Petri nets model," *Robot. Comput. Integr. Manuf.*, vol. 21, no. 6, pp. 550–558, 2005.

[13] C. Jung, H.-J. Kim, and T.-E. Lee, "A branch and bound algorithm for cyclic scheduling of timed Petri nets," *IEEE Trans. Autom. Sci. Eng.*, vol. 12, no. 1, pp. 309–323, Jan. 2015.

[14] R. Barreto, S. Cavalcante, and P. Maciel, "A time Petri net approach for finding preruntime schedules in embedded hard real-time systems," in *Proc. Int. Conf. Distrib. Comput. Syst. Workshops*, 2004, pp. 846–851.

[15] L. Ni, J. Zhang, C. Jiang, C. Yan, and K. Yu, "Resource allocation strategy in fog computing based on priced timed Petri nets," *IEEE Internet Things J.*, vol. 4, no. 5, pp. 1216–1228, 2017.

[16] S. K. Baruah, "Optimal utilization bounds for the fixed-priority scheduling of periodic task systems on identical multiprocessors," *IEEE Trans. Comput.*, vol. 53, no. 6, pp. 781–784, Jun. 2004.

[17] J. Goossens, E. Grolleau, and L. Cucu-Grosjean, "Periodicity of real-time schedules for dependent periodic tasks on identical multiprocessor platforms," *Real-Time Syst.*, vol. 52, no. 6, pp. 808–832, 2016.

[18] H. Baek, J. Lee, Y. Lee, and H. Yoon, "Preemptive real-time scheduling incorporating security constraint for cyber physical systems," *IEICE Trans. Inf. Syst.*, vol. E99.D, no. 8, pp. 2121–2130, 2016.

[19] R. Begam, X. Qin, D. Zhu, and H. Aydin, "Preference-oriented fixed-priority scheduling for periodic real-time tasks," *J. Syst. Archit.*, vol. 69, pp. 1–14, Sep. 2016.

HONGYU JIN received the B.S. degree in mechanical engineering from the Harbin University of Science and Technology, Harbin, in 2011, and the Ph.D. degree in mechanical engineering from the Harbin Institute of Technology, Harbin, in 2017.

Since 2017, he has been a Research Assistant with the Research Division of Numerical Control Technology, Harbin Institute of Technology. His research interests include open CNC systems and intelligent machining.

YANG WANG was born in 1960. He received the Ph.D. degree in mechanical engineering from the Harbin Institute of Technology, Harbin, in 1999.

From 1988 to 1995, he was an Assistant Professor with the School of Mechanical Engineering, Harbin Institute of Technology, where he has been a Professor with the Mechanical Engineering Department, since 2000. He has authored two books, more than 100 articles, and more than 30 inventions. His research interests include laser processing and aircraft manufacturing technology.

QIAN WANG was born in Hubei, China, in 1994. He received the B.S. degree in mechanical engineering from the Hefei University of Technology, in 2016, and the M.S. degree in mechanical engineering from the Harbin Institute of Technology, in 2018. He is currently engaged in wireless network with Huawei Technology Co., Ltd. His research interests include open CNC systems and virtualization technology like docker.

JIANKANG LIU received the B.S. degree in mechanical engineering from Huaqiao University, Xiamen, China, in 2011, and the M.S. degree in mechanical engineering from the Harbin Institute of Technology, Harbin, China, in 2013, where he is currently pursuing the Ph.D. degree. His research interests include real-time control system design and open CNC systems.

SHUHUA WANG was born in Guangzhou, China, in 1960. He received the B.S. degree in mechanical engineering from the South China University of Technology, Guangzhou, in 1982, and the Ph.D. degree from the University of Yamanashi, Japan, in 1996.

From 1982 to 1990, he was a Lecturer with the Department of Mechanical Engineering, and from 1984 to 1987, he was the Chief of the Mechanisms and Machine Theory & Robot Lab, South China University of Technology. From 1990 to 1991, he was a Visiting Research Fellow with the University of Yamanashi. He is currently the Director of the Innovation and Development Division, Foxconn Industrial Internet Co., Ltd. His research interests include industrial robot, the IoT, manufacturing intelligence, and industrial big data.

JUN ZHANG was born in Hanzhong, Shanxi, China, in 1973. He is currently the Leader of the Ultra-precision Technology Development Department, Innovation and Development Division, Foxconn Industrial Internet Co., Ltd. His research interests include ultra-precision manufacturing technology, ultra-precision equipment and core components, the IoT, and manufacturing intelligence.

SHANGHUA HAO was born in Henan, China, in 1982. He received the B.S. degree in computer science and technology from the Zhengzhou University of Light Industry, in 2005, and the M.S. degree in mechanical manufacturing and automation from Wuyi University, Guangdong, in 2009. He is currently an Engineer of the Innovation and Development Division, Foxconn Industrial Internet Co., Ltd. His research interests include open CNC systems, CAM technology, the IoT, and manufacturing intelligence.

HONGYA FU was born in Harbin, China, in 1962. He received the B.S. and M.S. degrees in mechanical engineering from the Harbin Institute of Technology, Harbin, in 1987.

From 1992 to 2001, he was an Assistant Professor with the School of Mechanical Engineering, Harbin Institute of Technology, where he has been a Professor with the Mechanical Engineering Department, since 2002. He has authored two books, more than 50 articles, and more than 20 inventions. His research interests include fiber winding molding technology, fiber automatic laying technology, and open CNC systems.