Design and implementation of partition Operating System based on ARINC653 standard

Yongchao Tao1,*, Qilong Song1 and Songhao Piao1
1 Shenzhen Academy of Aerospace Technology, Shenzhen, China

*Corresponding author e-mail: taoyongchao@chinasaat.com

Abstract. The ARINC series of standards are mainly used to define how to use the computing resources of avionics. The ARINC 653 standard applies to the service interface of the spacecraft's integrated electronic operating system to the upper-layer application software. It isolates the application software from the underlying processor architecture and meets the requirements of the on-board real-time system with good scalability. This article will follow the ARINC 653 standard, design and implement a partition operating system, providing basic system static configuration capabilities, space partition capabilities, time sharing capabilities, inter-zone communication capabilities, and security monitoring capabilities.

1. Introduction

As the requirements for satellite performance and functional density increase, the individual software on the satellite is gradually shifting to integrated electronic software, from the autonomous control of the sub-systems to partly or completely centralized management by the central unit, which increases the number of integrated electronic software interfaces and the task requirements Diversified and complicated information flow.

Especially after the integrated electronic software integrates the attitude orbit control software, the existing framework cannot meet these needs. It is necessary to introduce a complete set of operating system ideas to comprehensively manage various interface resources, schedule task execution, and coordinate message transmission between terminals[1].

In this paper, by studying and realizing on-board operating system specifications, software levels and interfaces can be established, and software componentization and layered design framework can be realized, thereby improving software reuse rate and engineering management efficiency in order to improve the efficiency of software development, testing, and management, the key is to increase the software reuse rate[2].

The ARINC 653 standard applies to the service interface of the spacecraft's integrated electronic operating system to the upper-layer application software[3]. It isolates the application software from the underlying processor architecture and meets the requirements of the on-board real-time system with good scalability[4].

This operating system standard will facilitate software portability, reusability, modularity, and coexistence of software with multiple levels of authority. The core idea is time-sharing and partitioning[5]. Through time-sharing, multiple applications can be isolated and run concurrently in time, and multiple applications can be isolated and protected in running space through partitioning. These are well in line with the requirements of high-reliability on-board software systems.
2. Functional requirements analysis

The partition operating system will follow the ARINC 653 standard and provide basic system static configuration capabilities, space partition capabilities, time sharing capabilities, inter-partition communication capabilities, and security monitoring capabilities.

(1) The system static configuration capability means that the user can statically configure the system attributes through the graphical interface, and the system generator generates the final runnable system that meets the user's needs.

(2) Space partition capability means that the system can support multiple partitions, can perform partition scheduling, and prevent illegal access to external partitions by processes in the partition.

(3) Time-sharing capability means that the system can perform partition scheduling at specified time points according to the user's configuration data, so that multiple applications can be isolated and run concurrently in time.

(4) Inter-zone communication capability means that the system can support inter-zone pipeline communication, and supports various forms such as sampling mode and queue mode.

(5) Security monitoring capability means that the system can support three-level security management, reporting and responding to errors and faults at the module level, partition level, and application level.

3. System analysis and design

In the partition operating system, the personnel involved in the final implementation of the system can be divided into system planners and application programmers. The system planners are responsible for static configuration of system properties, and through graphical configuration methods, the final configuration required by the system is generated. Application programmers develop applications in each partition according to system requirements.

This system can be divided into three parts: system configurator, system generator, and operating system core (system partition), as shown in Figure 1 below:

![Figure 1. Partition operating system structure](image)

The main function of the system configurator is to provide system planners with graphical static configuration capabilities of the system, including the number of system partitions, partition attributes, inter-partition communication pipeline settings, security monitoring settings, and so on. The generator will use Eclipse as the development platform, use GMF technology to provide users with graphical configuration capabilities, and automatically store user configuration data into XML files through GMF, and then use DOM technology to complete the analysis of XML files, based on the core data structure of the system Definition, complete the conversion from XML data file to the final configuration table of the system. Under the constraints of ARINC 653 standard, complete the static configuration of the system.
The system generator mainly compiles and links the system configuration table and other source codes together to generate the final executable file of the system to complete the final generation of the system. After that, the system generator will scan the target files of each partition, capture sensitive instructions (such as WRPSR-like privileged instructions, I/O instructions, etc.), and replace them with soft trap instructions for system partition management. Finally, the generator will link the target files generated by the core code of the operating system, the configuration file code, and the revised target files of each partition to generate the final executable file.

The operating system core module is the system partition, which mainly completes the realization of the core functions of the operating system. It mainly includes partition scheduling, partition memory protection, inter-partition communication, security monitoring, sensitive command execution, equipment management, etc. Partition scheduling ensures that each partition can be statically scheduled according to the system configuration to complete the normal operation of applications in each partition. Partition memory protection ensures that the programs in the partition can only operate on the memory range of the partition to which they belong, and cannot operate beyond boundaries. The inter-area communication guarantee system has inter-area communication capabilities that meet the system configuration, supports various types of pipeline operations, and ensures the correctness of communication. The security monitoring assurance system has security monitoring capabilities that meet the system configuration, can complete three-level security management, report and respond to errors and failures at the module level, partition level, and application level. Sensitive instruction execution will ensure the virtual execution of sensitive instructions in the partition application code, ensuring the correct execution of partition applications without affecting system behavior. Device management will ensure the system's control of all devices, that is, provide hardware access services for applications in the partition without affecting system behavior. With the support of the above three main modules, developers can complete the final system implementation. Among them, system planners statically configure system attributes through the system configurator, including partition attributes, port attributes, system security management attributes, etc.

Under the constraints of the configured system attributes, the application programmer completes the implementation of the applications in each partition, and completes the independent compilation and instruction replacement of the code in the partition through the system generator, and then through the support of the compiler and linker, the operation The system core code is compiled and linked with the revised target code in the partition to generate the final system executable file.

4. System analysis and design

4.1. System static configurator

In this system, a system planner needs to configure the minimum set of attribute information for the system as memory requirements, cycle, execution time, and port attributes. All configured information is stored in the form of system configuration information table. Similarly, a minimal set of system configuration information tables are system initialization configuration tables, inter-partition communication configuration tables, and security monitoring configuration tables.

Take the configuration of partition attributes as an example: after the user has configured the system to have several partitions, the attributes will be configured for each partition. The main configuration items are as follows:

1. Partition ID: Each partition has a unique digital ID. The operating system can use this ID to represent the partition and complete operations such as partition activation and message transmission.
2. Name: Each partition has a unique name.
3. Memory requirements: defines the memory boundary of the partition, and the appropriate code segment and data segment range.
4. Partition interval: defines the time interval at which the partition must be activated to ensure the operability requirements.
5. Partition duration: in a partition cycle, the execution time required by the partition.
6. Inter-zone communication requirements (ports): identification for inter-zone communication.
After the user configures the six basic attributes of a partition, the configuration of a partition graphic element is completed. GMF can save these configuration information as an XML file, and then the configurator will parse the XML file through DOM according to the data structure defined by the partition attribute of the operating system, assign the configuration data to the data structure, and complete the configuration from graphics to core data. Conversion of structural configuration.

```c
struct partitionAttribute {
    IndenterValueType      PartitionIdentifier;
    NameType              PartitionName;
    CriticalityType           Criticality;
    TimeType               partitionInterval;
    TimeType               persistentTime;
    NameType              PortName;
    ......
};
```

4.2. Partition mode switch

In the ARINC 653 standard, the most important is the concept of partition. Each partition can be regarded as an execution unit of application software. In this unit, all applications must meet the partition constraints of the unit. The operating system needs to provide robust partition management capabilities.

The current execution status of the partition represents its working mode. There are four partition working modes:

1. IDLE: In this mode, no application is executed in the partition and the partition is not initialized, but the partition time window in which the partition should be allocated is still reserved;
2. NORMAL: In this mode, the partition has been initialized and the process scheduler in the partition has been activated, all processes have been created, and are in a ready state to run at any time;
3. COLD_START: In this mode, the initialization phase of the partition is in progress, LOCK_LEVEL>0, preemption is prohibited, and related applications are executing their respective initialization codes;
4. WARM_START: In this mode, the initialization phase of the partition is in progress, LOCK_LEVEL>0, preemption is prohibited, and related applications are executing their respective initialization codes. This mode is similar to COLD_START, but its initialization environment may be different.

The transformation of each model is shown in Figure 2 below.

![Figure 2. Partition execution mode and its conversion](image)

4.3. Partition scheduling

Interval scheduling will follow fixed and cyclical rules. In order to ensure the cyclic execution of the system, the operating system must ensure a time scheduling framework consisting of a fixed execution period, which can be executed periodically and repeatedly during the operation of the entire system. The operating system activates the execution of each partition by allocating time partition windows, and each partition window is defined by its start time plus its expected execution time. In
the entire system, the execution sequence of the partitions is statically configured by the system planner and saved in the system configuration table to complete a definite partition scheduling process. Each partition uses processor resources at a pre-planned time.

![Partition scheduling example](image)

**Figure 3.** Partition scheduling example

The interval scheduling is completed by the scheduler, which is called the partition scheduler (PTSCH) for the time being. The function of PTSCH is divided into two parts, the first part is to select the partition to be operated, and the second part is to switch between partitions, including on-site protection and recovery. In order to ensure that PTSCH can control and maintain the operation of each partition, each partition has its specific attributes.

According to the user's static configuration, the partition scheduling sequence will be stored in the form of an array (also in the form of a secondary linked list). In PTSCH, n represents the total number of partitions that need to be run in a major period of time, then the array of the entire scheduling sequence can be defined as pID_SCH[n], each item in the array saves the partition ID, and curLocation represents the array where the currently running partition is located. Use NEXTPT to indicate the ID of the next running partition. When selecting a partition, you only need to do:

\[ \text{NEXTPT} = \text{pID}_{\text{SCH}}[(\text{curLocation}+1)\mod n] \]

It can realize fixed sequence and cyclic operation.

4.4. **Inter-zone communication**

The most basic base address for inter-area communication is the channel (CHANNEL). A pipeline can be defined as a logical connection from a data source to one or more destinations. At the same time, the data source or destination can be one or more partitions.

The most basic base address for inter-area communication is the channel (CHANNEL). A pipeline can be defined as a logical connection from a data source to one or more destinations. At the same time, the data source or destination can be one or more partitions.

The partition accesses the pipe through an access point called a port (PORT). A pipe can contain one or more ports. Each port provides all the resources needed for data exchange through a specific pipeline for the partition. At the same time, a partition can exchange data through multiple channels. A pipe can be described as a path connecting one sender and multiple receivers.

Each pipe corresponds to a section of accessible physical memory, and applications in the partition can write data to the memory through the port. The port is responsible for managing this memory. If it is a sampling port, simply overwrite the data to ensure that the latest data is stored in the memory; if it is a queue port, you need to maintain this memory and implement it as a queue. The form of FIFO or priority manages the sending or receiving of data.

Since partitions are running in rotation, only one partition is executing at any time, so there is no need for one party to finish writing and then receiving the communication pipeline between partitions. However, it is necessary to manage the situation where multiple applications in a partition compete for a port.
4.5. Security Monitoring

The safety management function is used to report and respond to errors and failures at the module level, partition level, and application level. It can prevent the spread of system errors and failures. The security management function of the system will be completed through the security management configuration form and an application-level error management process. The security management configuration table can be divided into module security management (module HM), multi-partition security management (multi-partition HM), and partition security management (partition HM tables). Different configuration tables will be retrieved for different error types to complete error response. Take different measures.

As can be seen from the above figure, when a process-level error occurs and the error management process can handle the error, the error will be handled by the error management process. Other situations are handled by a three-level security management form configured by the user.

For different types of errors, the following measures will be taken:

1) Process-level recovery measures: The main methods include ignoring, stopping the error process and restarting, stopping the error process and starting another process, stopping the error process, restarting the partition, and suspending the partition (IDLE);
2) Recovery measures at the partition level: The main methods include ignore, pause the partition (IDLE), and restart the partition;
3) System-level recovery measures: The main methods are ignore, shut down the system, restart the system, etc.

5. Experimental data and summary

Test hardware: BM3803 development version, 100Mhz frequency

Test plan: The system is configured with 4 partitions, and each partition runs 509 multi-task applications based on Xgc Ada, and each application contains 4 processes.

Test results: Fully meet the system's static configuration capabilities, spatial partitioning capabilities, time-sharing capabilities, inter-zone communication capabilities, and security monitoring capabilities. Performance indicators are as follows:

Table 1. Performance test results

| Numble | Result analysis |
|--------|----------------|
| 1      | Partition switching time is less than 500 microseconds |
| 2      | Interrupt response time is less than 20 microseconds |
| 3      | The maximum refresh rate is not less than 10Hz |
| 4      | Support the maximum message length not less than 256 bytes |
| 5      | Failure response time is less than 1 millisecond |

6. Acknowledgments

This work was financially supported by Shenzhen Special Fund for Strategic Emerging Industry Development JCYJ20170816151055158 and Key Basic Research Projects of the Military Science and Technology Commission Basic Strengthening Plan 2019-JCJQ-ZD-097-00.

References

[1] Bichpuriya Y K, Saxena A K, Kalra P K. Provision for reactive power market in the reactive power control groups[C]. in TENCON 2008 - 2008 IEEE Region 10 Conference, 2008:1-4.
[2] Wachs M, Abd-El-Malek M, Therese E, et al. Argon: performance insulation for shared storage servers[C]. in FAST, 2007:5.
[3] Shue D, Freedman M J, Shaikh A. Performance Isolation and Fairness for Multi-Tenant Cloud Storage[C]. USA: 2012.
[4] Mccullough J C, Dunagan J, Wolman A, et al. Stout: An adaptive interface to scalable cloud storage[C]. in USENIX Annual, 2010., 2010:101.
[5] KargerPA, affordDR. I/OforVirtual MachineMonitors: SecurityandPerformance Issues[C]. Security & Privacy, IEEE. 2008, 6(5): 16-23.