A mission oriented reconfiguration technology for Spaceborne FPGA

J X Qin, J Yang*, Z Qu and Y X Wang
College of Artificial Intelligence at National University of Defense Technology, Changsha, Hunan, China.
Email: iohn323@163.com

Abstract. Reconfigurable technology based on FPGA is increasingly used in satellites to meet the demand of the increasing missions that satellites need to complete in recent years. In this paper, a mission oriented reconfiguration technology for spaceborne FPGA is introduced to address the problem that how to meet the demand of satellite reconfiguration mission to orbit, execution time and FPGA resource requirements. Satellite reconfiguration system is divided into three parts, that is, mission management module, reconfiguration management module and reconfigurable FPGA. Firstly, a mission list is set up to record the mission's execution time and space attributes, hardware space attributes and priority attributes. The mission is allocated to the FPGA resource area correspondingly based on the flexible region model. According to the time attribute and priority attribute of the mission, the reconfiguration management module carries out the reconfiguration action. And the reconfiguration management module mainly deals with reconfiguration library management and the entire or partial reconfiguration according to the configuration file. After the reconfigurable bit files are loaded to FPGA, the transformation and execution of the hardware resources from the mission to the hardware are completed. Finally, an evaluation of the real-time performance, reliability and security of the design is carried out. Experiments that based on a verification system composed of raspberry/PI and ZYNQ are conducted and results fully prove that the design in this paper can support mission oriented spaceborne FPGA reconfiguration.

1. Introduction
The payload of traditional satellite is solidified with a poor flexibility. For the reason that the satellite can only play a role in the specific orbit, and the other time, which takes up the most part, is in the closed state, resulting in an idle and waste of the satellite resources. The application of reconfigurable technology to satellites is a good way to solve this problem. While the reconfigurable technology in the ground is more mature, it has only started in recent years for satellites. A reconfigurable Satellite TT&C (telemetry, track and command, TT&C) communication system based on FPGA is built in [1]. Its goal is to realize the real-time update of the small satellite communication load function and the switch of communication mode through the configuration of the ground communication station. The software defined satellite "quantum" project, conducted by the ESA (European Space Agency, ESA), the European Communication Satellite Corporation and the Airbus defense and Aerospace Corp, which is expected to be launched in 2019, use the FPGA to reconfigure the function of the payload as in [2]. However, the reconfiguration of the satellite is based on the specific mission requirements. The FPGA resource requirements of the mission, the track and time of the mission execution are different, and the priority is also different. How to map missions to specific hardware resources, and complete...
reconfiguration according to the needs and characteristics of satellites, there are few papers relatively on this topic. Nevertheless, the reconfiguration technology on the ground is more mature. There are many researches on mission placement. For example, a reconfigurable hybrid mission scheduling algorithm is introduced to reduce configuration time in [3]. Reference [4] discusses several reconfigurable resource management models, and proposes a one dimensional online scheduling model, which focuses on mission scheduling. In order to solve reconfigurable computing, mission scheduling system based on spatio temporal context is studied in [5]. Considering the actual situation of the satellites, the mission oriented FPGA reconfiguration technology is studied in this paper.

The missions of satellite will be more and more. Using reconfiguration technology to execute multiple missions can effectively save resources and improve resource utilization. Satellite reconfigurable systems are mainly divided into two parts: satellite to ground reconfiguration loop and satellite reconfiguration system [6]. This paper only considers satellite reconfiguration system, and mainly solves the problem that how to map the satellite mission to the hardware resources, and how to execute multiple reconfiguration missions. The work based on this paper can not only support the reconfiguration of the satellite borne FPGA, but also enable the satellite to gain a kind of certain autonomy, and can satisfy some special users like in the overseas and the public seas that are inconvenience of controlling the satellites, and even support the sudden reconfiguration of the real-time updating application scene.

2. Mission oriented FPGA reconfiguration system

FPGA reconfiguration system, in this paper, specifically refers to modules containing FPGA resources, not a specific load or single chip. However, in order to facilitate the experiment, this paper will consider the selection of FPGA, which will be explained later. Mission oriented refers to reconfiguration activities centered on missions and aims at achieving the goals of missions, rather than reconstructing them solely for reconfiguration. The mission of the satellite should be dynamic and diverse. For specific track position, in specific time period and specific application scene, the mission is different, and the reconfiguration also has the corresponding change. For example, when the satellite moves over the land and sea, the remote sensing algorithm is different; the detection algorithm is different from the different seasons over the same land.

In this paper, the spaceborne FPGA reconfiguration system is divided into three parts, that is, on-board computer mission management module, reconfiguration management module and reconfigurable FPGA. As in figure 1, a list of missions is set up for satellites, recording the hardware space properties of each mission, executing space-time attributes and priority attributes. According to the hardware space attribute, the mission is placed to the hardware resources based on the region model. The mission reconfiguration time is determined according to the execution time attribute and priority attribute. The reconfiguration management module manages reconfigurable bit files, and the reconfigurable bit files are loaded into reconfigurable FPGA to complete the reconfiguration mission according to the configuration file.

![Figure 1. Mission oriented spaceborne FPGA reconfiguration system.](image-url)
2.1. Mission management
Satellite On-board computer mission management module is responsible for managing satellite reconfiguration missions. The mission referred in this paper specifically refers to the mission of reconfigurable FPGA, because not all missions need FPGA for reconfiguration. A mission list is set up for the reconfigurable missions carried by the satellite, and the hardware space attributes, space-time attributes and priority attributes are registered. According to the hardware space properties of reconfiguration missions, the first step is to put reconfiguration missions into specific FPGA resource blocks. As shown in figure 2, there are three reconfigurable resource model diagrams, followed by one-dimensional model map, as in (a), two-dimensional model map, as in (b) and regional model map [4] as in (c). The peripheral bounding box represents the FPGA resources that can be reconfigured, and the small rectangles represent the FPGA resources occupied by the mission. The one-dimensional model is first proposed by [7], which simplifies placement and scheduling, and is suitable for Xilinx Virtex implementation. However, this model will lead to fragmentation inside and outside the FPGA, and need to adopt defragmentation technology. It is first proposed in [8] that the reconfigurable surface is divided into a static and fixed number block in a two-dimensional model. Each block can hold one mission at a time. This area partition simplifies placement and scheduling, and is conducive to online addressing. But its drawback is internal fragmentation. When a mission requires less than one block of resources, other resources in the block are wasted.

![Figure 2. Reconfigurable resource model diagram.](image)

In this paper, region model diagram is used based on the method proposed by Busquere J P et al in [9]. This method is the most flexible and can be placed in any size resource for the size of the mission requirement resource, so the resource utilization rate is high. It only needs the reservation space as the communication channel. According to the resource requirements of specific reconfiguration missions, specific areas can be delineated, and then loaded and run in real time. Moreover, when we need to update the overall layout, we can use entire reconfiguration to achieve overall resource reconfiguration. But entire reconfiguration needs to interrupt part of the program's running, so it is good advice to try to use partial reconfiguration.

Reconstructing the temporal and spatial attributes of a reconfiguration mission is the question of where the mission’s executed orbit is and whether it is executed after it reaches the area. Considering the two attributes of reconfiguration mission, we can sort the mission and control and execute the reconfiguration mission according to sorting. Mission management divides missions into current missions, planning missions and appointment missions. The current mission is a running mission in FPGA. There can be one or more missions, with multiple current missions because FPGA can be divided into multiple reconfiguration areas and support multiple current mission parallel processing. Planning missions are missions that are sorted in the mission list in order of execution. The appointment mission is the mission to be reconstructed and implemented next. It also allows multiple appointment missions under the support of FPGA resources.

2.2. Reconfiguration management
Reconfiguration management module is mainly used for reconfigurable bit file library management and reconfiguration control. In the ground system, FPGA reconfiguration in the system can be easily realized by replacing the program storage chip directly or using the erasable memory for offline/online programming. But in satellite, the reconfiguration of FPGA program needs some special restrictions, such as the non disassembly of the equipment after launching, the memory chip to resist space irradiation, the limited communication channel resources from the ground to the satellite. To store reconfigurable bit file libraries and to ensure the correctness and reliability of reconfigurable bit files, configuration data and configuration process must be a key concern for satellites.

The reconfigurable bit file can be stored in advance or later uploaded by the ground control station. All reconfigurable bit files are stored in the highly reliable memory of the on-board computers to resist space irradiation. In order to ensure the reliability of reconfiguration, satellites should avoid the possibility that executing reconfiguration while receiving the reconfiguration bit file. As shown in table 1, the scale of FPGA commonly used in aerospace is 3 million gates [10]. Calculated according to the commonly used uplink remote code rate 4000bit/s, the transmission time takes 2900 seconds. Such a long time is very prone to errors in the process of reconfiguration. Therefore, we need to upload first and then reconfigure. When the reconfiguration is started, this paper adopts the way that the spaceborne computer is transmitted to DDR through the network port, and then is written directly to FPGA through ARM control. Because the reconfiguration control of FPGA requires the coordination of ARM. The on-board computer is only responsible for the management of reconfigurable bit files and configuration files. The actual control of FPGA file writing and boot startup requires ARM control.

| Table 1. Configuration data size and transmission time of FPGA used in spacecraft [10]. |
|---------------------------------|---------------------------------|----------------|----------------|----------------|
| Type of FPGA                      | Scale/ten thousand gates | Amount of the configuring data/bit | Storage space/Mbit | Transmission time/s |
|---------------------------------|----------------|----------------|----------------|----------------|
| XQVR300                         | 30            | 1751808        | 16             | 500            |
| XQVR1000                        | 100           | 6127744        | 16             | 1500           |
| XQR2V3000                       | 300           | 10494368       | 16             | 2900           |
| XQR2V6000                       | 600           | 21849568       | 32             | 5900           |
| XQR4VSX55                       | 550           | 22744832       | 32             | 6000           |
| XQ5VFX130T                      | 1300          | 49234944       | 49             | 13000          |

In this paper, we verify the integrity and correctness of reconfigurable bit files. Table 2 is the structural diagram of the reconfigurable bit file, and table 3 is the first structure diagram of partial reconfiguration. As can be seen from the structure, the reconfiguration bit file can have one or more partitions corresponding to the partial reconfiguration. The Header Authentication is a validation of the header and can be selected. In each Partition, Checksum is used to check the files. After uploading a file, it is necessary to check whether it meets the required header and tail before loading it. The header is different or incorrect, indicating errors in the transmission and storage process. The tail is different or incorrect, maybe because there is no transmission to complete, or other problems occur. If the check is not passed, reconfiguration is not allowed.

| Table 2. Structure of BOOT.bin [11]. |
|-----------------------------------|-----------------------------------|----------------|----------------|----------------|----------------|
| Boot Image Header                 | Register Initialization            | Image Header Table | Partition Header Table | Header Authentication (Optional) | Encrypted FSBL | FSBL RSA authentication (Optional) | Partition ... |
|-----------------------------------|-----------------------------------|----------------|----------------|----------------|----------------|----------------|-----------------------------------|
|                                   |                                   |                 |                 |                 |                 |                 |                                    |
Reconfiguration management is responsible for executing the reconfiguration action to the reconfigurable hardware. For FPGA to complete a reconfiguration, we need to consider its reconfiguration type, file loading, boot and reset power on. The reconfiguration types are divided into entire reconfiguration and partial reconfiguration. Entire reconfiguration is a redistribution of the whole resources of FPGA, which will affect the current running program. Partial reconfiguration can achieve partial reconfiguration without affecting the normal operation of other functions. The boot and reset power on are designed for specific hardware models and requirements. The key to reconfigure FPGA's logical resources is to reorganize its combinational logic. In the FPGA design of reconfigurable hardware units, the PROM/FLASH is configured to store FPGA software and hardware design files, and then by receiving or prestoring the design files, after receiving the reconfigured instructions, the reset signal is sent by the configuration management module, the FPGA power is restarted and the bits are reloaded into the FPGA, and the logical resources are reloaded. All refurbishing is carried out to achieve entire reconfiguration. The partial reconfiguration does not require power break. By addressing, the bit files that were reconfigured are loaded into the FPGA by ARM control to realize the reorganization of the logical resources in this part of the region.

In order to achieve partial reconfiguration of the function after confirming the FPGA resource area of the mission, we need to design the interface at the same time. This paper is based on ZYNQ, and the structure of partial reconfiguration is shown in figure 3. A Reconfigurable Partition (RP) is opened up in the FPGA resource of ZYNQ, and the RP is an instantiation of reconfigurable functional parts, and the Reconfigurable Module (RM) is distributed in the region. A RP can have one or more RM in it. When the RP is reconfigurable through the AXI4-Lite bus and the control instruction is sent, the other programs outside can continue to run in parallel and will not be interrupted by this part of the reconfiguration. The control of this part is completed by the configuration management module of ARM and the on-board computer.

**Figure 3.** Architecture of partial reconfiguration [12].

### 3. Experimental verification

In this paper, an entire reconfiguration scenario and a partial reconfiguration scenario are set up to simulate two missions to verify the scheme. The first mission is ADSB, and the other mission, Add
and Product represents the signal process mission. Assuming that the satellite is a Leo 500km satellite, it firstly carries the ADSB function to detect aircraft flight data. ADSB refers to the broadcast automatic correlation monitoring technology, which can provide aircraft flight position and flight status. Then, when the X area is reached, it is no longer necessary to detect the aircraft, but rather need to turn it into a signal processing monitoring module. Therefore, it needs to be reconfigured into a signal processing module. When running to the Y area, the signal processing module changes, but it does not need to reconfigure all of it, only needs to change the processing method, so it only needs to reconfigure the signal processing part.

3.1. Mission management

The formula of orbital period is shown in (1), in which $T_s$ is the orbital period, $R_e$ is the earth's radius, $h$ is the orbit altitude, $G$ is the gravitational attraction constant, and $M$ is the mass of the earth:

$$T_s = \left(\frac{(R_e+h)^3}{GM}\right)^{1/2}$$

The satellite with a height of 500km has an orbital period of about 95 minutes. According to the experience in [6], the transit time only lasts for a few minutes. But it is enough to complete the mission and upload the configuration files in that time. Assuming that the reconfigurable bit file is stored in the file library, it is verified as the correct file. The following is a list of missions. The hardware area, execution time and space attributes and priorities of the mission are determined by the ground and sent to the satellite by the ground control station.

The area where the reconfigurable bit file was loaded is determined by the configuration file by addressing. A diagram is a specific address correspondence as in table 4. Through the communication channel described above, the reconfiguration mission can be mapped to the actual FPGA resource under the conditions of the presupposed address, and the control and interaction required for the mission, such as reading and writing data, will be completed. The size of the file in the table can also clearly distinguish entire reconfiguration and partial refactoring as in table 5.

| Mission | Address   | Length(hex) |
|---------|-----------|-------------|
| ADSB    | 0x10000000| 0098 9680   |
| Add     | 0x02000000| 3650        |
| Product | 0x01000000| 3650        |

3.2. Configuration and implementation of reconfiguration

ZYNQ, the Xilinx Virtex series, is chosen in the paper. Its FPGA chip is a series of FPGA chips which can be implemented by Xilinx to realize dynamic partial reconfiguration. And the FPGA chip supports the application of space class, so this paper uses it to do experimental verification. ZYNQ belongs to ZYNQ-7000 series, and it adopts ARM+FPGA architecture. After system reset and successful startup, the first running software is BootROM (no disk startup ROM), loading and running FSBL (First Stage Boot Loader, FSBL) and user code to complete the system initialization configuration. The control of FPGA is done by PS, so after receiving the reconfiguration command of the FPGA, boot the bit file by the PS and implement entire or partial reconfiguration [13].

Two projects are designed in this paper. The first is the ADSB mission, and the second is the signal processing mission which can realize partial reconfiguration that has stated above. The satellite carries ADSB, which can monitor the aircraft in a wider range and faster. When running to the area where signal processing is needed, the ADSB function is reconfigured to perform signal processing missions. This partial reconfiguration can be applied to the update of signal processing algorithm and partial modification of signal modulation system.

For a specified part of the reconfiguration area, as long as the interface is designed, and the number of FPGA resources is enough in this area, we can design updates to any required processing. For example, the signal processing algorithm product module, which is designed in this paper, is a
reconfigurable module RM with two input signals and an output signal, which can be designed as a required algorithm in RM. This consideration is very practical for satellites. For example, for signal processing, the interface generally does not change much, but the processing algorithm will need upgrading after a period of time. The communication system will change in 3 or 5 years or others. As shown in figure 4, this article uses MATLAB/Simulink to carry out a design of addition and production. Through this tool, the automatic generation of Verilog code can be completed quickly and the reconfiguration of the two algorithms is completed. It is similar to the two orthogonal signals of I and Q paths in the BPSK signal. At this point, signal synthesis or orthogonality is completed, and then output. This is a design for partial reconfiguration. The design can complete part of the function replacement, can be used for software upgrading, interface to reduce power and so on.

3.3. Assessment
The assessment of real-time is divided into entire reconfiguration and partial reconfiguration. The first is entire reconfiguration. When the satellite function requires completely reconfiguration, it is usually the time to upgrade the satellite function or when the malfunction occurred and the satellite needs to be reconfigured. In that case, the satellite does not require very high real time. It should be noted that this paper does not consider the upload time of reconfigurable bit files. To calculate the time needed to upload files to satellites, we should consider the capacity and speed of the satellite link, and so on. This is not the factor of reconfiguration. The design scheme of this paper opens up a specific storage area for reconfiguration bit file, waiting for the storage of the bit files to be uploaded and checking the checksum. Only to make sure the reconfiguration bit file is correct, does the reconfiguration instruction will be sent and the satellite functions are entirely or partially reconfigured. Because the interruption of FPGA in the middle will affect the operation of the program, so the measurement of time is relatively rough. The reconfigurable bit file the paper use is 4261 KB and the time that from sending the command to completing the entire reconfiguration is about 5 seconds. After analysis, transferring the reconfigurable bit file from the on-board computers to the DDR takes up the most part of time.

Table 6 shows the average time the two partial reconfiguration (Product and Add) taken from receiving the reconfiguration instructions to completing the partial reconfiguration. The test method uses the PMU (Performance monitor unit, PMU) in the ARM kernel to implement [14]. For partial reconfiguration in real time, compared with entire reconfiguration, it has higher real-time requirements. This is determined by the different applications. In the case of partial reconfiguration, in addition to the fact that the requirements for the update of the function may not be very high in real time, other changes such as the replacement of the software radio wave form processing and the transformation of the I/O type are both a strong real-time transformation. As shown in table 6, we can see that partial reconfiguration time is very small, at 100 μs level, and it can be used in most occasions.

As for reliability, the paper discussed above is concerned in some ways. First of all, the on-board computer ensures the reliability of satellite to ground transmission. Then, the reconfigurable bit file library is stored in the storage area which is resistant to space irradiation, and the files all are checked and verified. When loading the file, firstly pass it to the DDR, then after the verification, do the transmission and then load it to FPGA. The "readback" technology can be used to compare the configuration data, all register contents, and storage numbers [15]. If FPGA resources are enough, three times redundant backup can be carried out, and vote circuit also can be used to make a judgement to improve the reliability.
Figure 4. Partial reconfiguration to realize the upgrade of partial function.

Table 6. Real time test.

| RM | Product | Add |
|----|---------|-----|
| Time/μs | 1914 | 869 |

4. Conclusion
In this paper, a mission oriented spaceborne FPGA reconfiguration technology is introduced. Based on the region model, the problem of transformation from the reconfiguration mission to the execution of the hardware resources is solved. Not only the allocation and use of the global or local resources of the spaceborne FPGA can be realized through the technology of entire or partial reconfiguration, but also the use of FPGA can be effectively improved. Thus, the flexibility and autonomy of satellites can improve greatly. By assessing the real-time, reliability and security of the technology, it shows that this design scheme can be effectively used in mission oriented spaceborne FPGA reconfiguration.

References
[1] Y D Wen 2016 Research and design of mobile communication system based on Software Radio. Unpublished undergraduate dissertation, Hunan University, Changsha
[2] Eutelsat-Quantum : An overview of the European quantum satellite project http://www.360doc.com/content/17/0531/05/22081874_658600941.shtml
[3] S Shen, Z Y Zhu, J Wu 2014 Reconfigurable hybrid mission scheduling algorithm. Computer application. 32(2) L387–390
[4] M Jie, J X Wang 2018 A reconfigurable resource management model and its scheduling technology. Computer engineering and Applications. 7 L 245–250.
[5] C P Xu, Y H Chai, W C Huang 2015 Reconfigurable high performance mission scheduling system based on spatio-temporal context. Laboratory research and exploration. 3 L116–121
[6] G liang, W B Liang, H J Liu 2010 Design of reconfigurable communication system for low orbit satellite. Journal of aerospace.31(3) L185–191
[7] Brebner G, Diessel O 2001 Chip-based reconfigurable mission management. Proceedings of the 11th International Workshop on Field-Programmable Logic and Applications. 2147 L182–191
[8] Marescaux T, Andrei B, Dideriek V, Serge V, Rudy L 2002 Interconnection networks enable fine-grain dynamic multi-missioning on FPGAs. Proceedings of the 12th International Conference on Field-Programmable Logic and Applications. 2438 L795–805
[9] Busquere J P, Duearouge B, Grenier K 2005 Above IC MEMS capacitors for integrated reconfigurable circuits. Microwave Conference Proceedings. L4
[10] P Pang, W N Hao, W F Zhang, Y Xu, J B Zhu 2017 An engineering implementation scheme for
SRAM-FPGA in orbit reconfiguration. Spacecraft Engineering, 26(5) L51–6

[11] Xilinx, Inc 2015 Zynq-7000 All Programmable SoC Software Developers Guide

[12] Xilinx, Inc 2015 Reconfiguring Processor Peripheral Lab

[13] J H Lu, Z Jiang, M Ma 2013 Guidelines for hardware and software co design of embedded systems: Based on Xilinx ZYNQ (Beijing: China Machine Press)

[14] How to calculate the execution time of a certain program in SDK: https://blog.csdn.net/wanruoqingkong/article/details/37759015.

[15] L Fang, S J Wu, L Min, Y S Gong 2014 Translate. FPGA security design guide (Beijing: China Machine Press)