Frontiers of Information Technology & Electronic Engineering
www.jzus.zju.edu.cn; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

A spaceborne advanced storage system for remote sensing microsatellites*#

Shilei TU†1,2, Huiquan WANG†‡1,2, Yue HUANG1,2, Zhonghe JIN1,2

[1]*Micro-Satellite Research Center, Zhejiang University, Hangzhou 310027, China*

[2]*Key Laboratory of Micro-Nano Satellite Research, Zhejiang Province, Hangzhou 310027, China*

†E-mail: 12024046@zju.edu.cn; hqwang@zju.edu.cn

Received Oct. 9, 2022; Revision accepted Mar. 14, 2023; Crosschecked Mar. 1, 2024

Abstract: With the development of satellite miniaturization and remote sensing, the establishment of microsatellite constellations is an inevitable trend. Due to their limited size, weight, and power, spaceborne storage systems with excellent scalability, performance, and reliability are still one of the technical bottlenecks of remote sensing microsatellites. Based on the commercial off-the-shelf field-programmable gate array and memory devices, a spaceborne advanced storage system (SASS) is proposed in this paper. This work provides a dynamic programming, queue scheduling multiple-input multiple-output cache technique and a high-speed, high-reliability NAND flash controller for multiple microsatellite payload data. Experimental results show that SASS has outstanding scalability with a maximum write rate of 2429 Mb/s and preserves at least 78.53% of the performance when a single NAND flash fails. The scheduling technique effectively shortens the data scheduling time, and the data remapping method of the NAND flash controller can reduce the retention error by at least 50.73% and the program disturbance error by at least 37.80%.

Key words: Microsatellite; Spaceborne advanced storage system (SASS); Scalability; Performance; Reliability
https://doi.org/10.1631/FITEE.2200445 **CLC number:** V446+.9

1 Introduction

Recent advances in satellite miniaturization technology have directed efforts in remote sensing towards microsatellite constellations. In particular, the application of compact commercial off-the-shelf (COTS) components breaks through the limitations of microsatellites in terms of size, weight, and power (SWaP) consumption, and computing and storage performance (Radhakrishnan et al., 2016; Wang LG, 2019). As a result, microsatellite constellations have the advantages of low cost, short manufacturing cycle, small volume, quick response (Ding et al., 2020), and high revisit (Sezer et al., 2017) compared with conventional satellites.

As the core of remote sensing missions, remote sensing payload technology is developing rapidly, including real-time processing (Zhang Y et al., 2020; Zhou et al., 2020) and multipayload information fusion technology (Huang et al., 2012; Vakil et al., 2021). These technologies bring an explosion in the volume of remote sensing data, and a spaceborne storage system with high performance is urgently necessary.

One of the key parts of spaceborne storage systems, the storage medium, has experienced three development stages, represented by static random access memory (SRAM) and dynamic random access memory (DRAM), synchronous dynamic random access memory (SDRAM), and NAND flash (Sasada et al., 2005; Stuffler et al., 2007; Reid and Ottman, 2014). At present,

‡ Corresponding author

* Project supported by the Fundamental Research Funds for the Zhejiang Provincial Universities, China (No. 2021XZZX038)

Electronic supplementary materials: The online version of this article (https://doi.org/10.1631/FITEE.2200445) contains supplementary materials, which are available to authorized users

ORCID: Shilei TU, https://orcid.org/0000-0003-1647-4975; Huiquan WANG, https://orcid.org/0000-0003-0113-8223

NAND flash is the first choice for spaceborne storage systems (Fabiano and Furano, 2013; Song, 2015; Xu W et al., 2017; Liu GH et al., 2018; Zhang XY, 2020).

Based on the NAND flash array, many manufacturers are developing spaceborne solid-state records (SSRs) around the world, such as Astrium (Airbus Defence and Space, 2022), SEAKR (SEAKR Engineering, 2022), Calculex (Calculex, 2022), and Space Micro (Space Micro, 2022). Chinese research institutes and universities have also carried out research on large-capacity SSR, developing five generations in total (Li S, 2015; Zhu et al., 2020).

However, today's miniaturized spaceborne high-resolution cameras can reach 45 megapixels or more with 5 frames/s (1717 Mb/s). Calculex CSR-6602 can meet this requirement with its 3.2 Gb/s maximum write rate, but it is too heavy for microsatellites and consumes nearly 100 W (Calculex, 2022). Even a typical miniaturized SSR such as the one from Xidian University's spaceborne SSR cannot meet the demand (Li S, 2015) despite having a high-speed, large-capacity storage device (with a write rate of 951 Mb/s and a total capacity of 2 Tb).

Achieving high-performance, low power consumption miniaturized SSR with COTS devices is a win–win solution. However, SSR's COTS devices inevitably reduce the system reliability, and the use of various COTS payloads also challenges SSR's compatibility design. A spaceborne SSR with high scalability and reliability is essential in space science and technology development as the number of microsatellites expands, complex requirements increase, and development cycles shrink.

To solve the above problems of remote sensing microsatellite storage systems in terms of scalability, performance, and reliability, this paper proposes corresponding solutions after specific analysis. It proposes a spaceborne advanced storage system (SASS) for remote sensing microsatellites. Then a series of experiments are conducted to validate the efficiency of the proposed scheme and evaluate its performance and reliability. The advantages of SASS are summarized as follows:

1. Modules of SASS, the multiple-input multiple-out (MIMO) cache and the NAND flash controller, have excellent scalability to accommodate different missions and payloads. The NAND flash controller provides smaller configurable functional particles for greater system flexibility and robustness.

2. The performance of SASS is strong, and the dynamic programming method and queue scheduling method can effectively improve the efficiency of data scheduling to meet the requirements of high-speed multichannel data storage.

3. The proposed data remapping method can reduce the retention and NAND flash program disturbance errors. It partially compensates for the limitations of the error correction algorithm, improves the reliability, and extends the system's service life.

2 Concerns and solutions of the storage system

2.1 Scalability

High system scalability, which aims to accommodate different payloads and missions, is conducive to shortening microsatellite development cycles and technology iteration periods.

This paper's SASS architectures and modules revolve around modularity, standardization, loose coupling, and controllability. The architectures are open to users, and users can define system function and structure by software, supporting the business expansion of a single satellite and the capability improvement and rapid response of the constellation (Liu ZX, 2021).

2.2 Performance

The performance of processors and memories determines the upper and lower limits of the spaceborne storage system. Using a central processing unit (CPU), digital signal processor (DSP), graphical processing unit (GPU), or field-programmable gate array (FPGA) for storage systems and real-time processing is mainstream. However, due to limited computational resources, DSP and CPU are not suitable for real-time processing of remote sensing microsatellites, and GPU is also not suitable because of its high power consumption. FPGA has irreplaceable advantages in terms of on-chip resource storage, computing capabilities, and reconfigurable characteristics. It can meet large throughput requirements and real-time processing under

spaceborne conditions with lower power consumption (Yang Z and Long, 2015; Schmidt et al., 2017).

For memory, the preparation time of storage media will extend the operation time and reduce the ratio of the expected operation time to the actual operation time, that is, the writing efficiency (Wang GQ et al., 2021). The higher the writing efficiency, the shorter the writing time, which means the higher the real-time performance.

Double-data-rate-3 SDRAM (DDR3 SDRAM) is the storage medium of the MIMO cache. When writing n bytes of data, the time consumed is denoted by t_{WRITE}:

$$t_{\mathrm{WRITE}} = t_{\mathrm{RCD}} + t_{\mathrm{CWL}} + nt_{\mathrm{CK}} + t_{\mathrm{WR}} + t_{\mathrm{RP}}, \quad (1)$$

where t_{RCD} is the delay time from the active command to read or write, t_{CWL} is the delay time from the write command to the first data transmission, t_{CK} is the working clock of DDR3, t_{WR} is the minimum interval from the last data transmission to the precharge command, and t_{RP} is the minimum interval from the precharge command to the next active command (Micron, 2021).

Each row has 1024 cells in a typical DDR3, and the maximum writing efficiency can be calculated when the write burst length is set to 8:

$$\eta_{\mathrm{WRITE}} = \frac{nt_{\mathrm{CK}}}{t_{\mathrm{WRITE}}},\ 0 < n \le 128. \quad (2)$$

To improve data scheduling efficiency, the adaptive dynamic programming (ADP) algorithm is introduced in this study. In essence, it approximates the optimal control strategy in dynamic programming by using the reinforcement learning principle (Werbos, 2011; Kong et al., 2021). At present, the ADP structure has been applied in missile guidance, automatic driving, robot balance control, and other problems (Xu X et al., 2007, 2017).

For NAND flash, the array design expands the data bit width and increases the data throughput. However, in a regular design, the smallest unit of the system is the entire array, and the damage to a single NAND flash chip of the array can have a severe impact on the system. Smaller configurable functional particles are introduced into the architecture to enhance the robustness and flexibility of the system.

Moreover, the programming time of the NAND flash reduces the writing efficiency. The pipeline operation addresses this issue.

2.3 Reliability

In space missions, the storage medium is easily affected by the single particle effect, especially the single event upset (SEU). In a conventional design, triple modular redundancy (TMR) and error correction code (ECC) are used to address such random errors (Yin, 2018).

Common ECC algorithms in the NAND flash storage system include the Hamming algorithm (Rhee et al., 2010), the Reed–Solomon (RS) algorithm (Xiao et al., 2011), and the Bose–Chaudhuri–Hocquenghem (BCH) algorithm (Mahdy et al., 2020). Both RS and BCH algorithms can correct multibit errors, but they require more system resources and the codec process is more complicated. The Hamming code provides the easiest hardware implementation.

Wear balancing (Yang CL, 2008), file systems (Dong, 2017), and bad block management (Wei et al., 2016) are also key and conventional technologies in storage systems. However, a more reliable design must be considered due to the physical properties of NAND flash, which is a nonvolatile memory based on a floating gate transistor structure. When the negative electron is injected into or removed from the floating gate under the action of the control gate, the storage state of the NAND single transistor changes, as per programming (1 to 0) or erasure (0 to 1) (P/E) operation. The number of operations of negative electronics is reduced, so are the erasure and programming times (Pavan et al., 1997).

This structure's common types of data errors are program disturbance errors and long-term data retention errors because of electronic leakage and programming page to nonprogramming page interference (Cai et al., 2013). The single-level cell (SLC) NAND flash has far fewer program disturbance errors than the multilevel cell (MLC) and trinary-level cell (TLC) NAND flash. Nevertheless, data retention errors will increase in applications such as aerospace, where electrons can experience thermal motion due to energy radiation (Wang JX, 2010).

As the number of P/E operations increases, the bit error rate (BER) grows exponentially. Then it exceeds

the limited error correction capability of the ECC, meaning that the storage system will reach the end of its service life without on-orbit maintenance services. This paper proposes a data remapping method to reduce the BER of data and extend the system service life.

3 Implementation of SASS

3.1 Highly scalable architecture

SASS consists of a MIMO cache and a NAND flash controller, as shown in Fig. 1. The two modules are loosely coupled and communicate only via a gigabyte transceiver X (GTX). The integrated electronic system of the microsatellite is the upper computer of SASS.

3.1.1 Modular and standardized MIMO cache

The hardware structure of the controller is shown in Fig. 2. Following the concept of hierarchical system design, the functional modules of the MIMO cache are encapsulated into different layers of standard modules to provide different services. There is a low coupling degree between layers, and the key parameters of each layer can be configured by users. The software of each layer is easily developed, replaced, and tested independently.

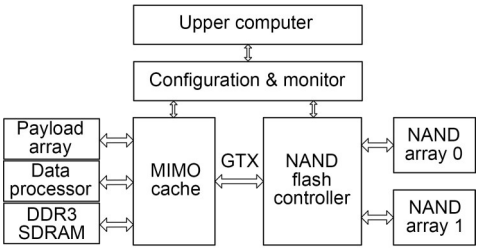

Fig. 1 Architecture of the spaceborne advanced storage system

This design has the advantages of standardized interfaces and services, high reusability, and good testability. The structure consists of a logical layer, transport layer, physical layer, configuration fabric, clock network, reset network, monitor, and input/output (I/O) first-input first-output (FIFO).

The logical layer is the policy layer for data exchange. To achieve real-time data distribution, the cache modes of the switching unit include the input cache, output cache, and shared cache. To increase the switching capacity, support a higher line rate, and maximize DDR3 efficiency, combined with the priority characteristics of input and output data, the output cache mode is adopted in this paper (Zhang Y et al., 2015). This layer schedules input data from user input FIFOs, sends output data to FIFOs, and generates action instructions for the transport layer.

The transport layer provides routing information based on address configuration, instructions, and data frames during transmission. Then it implements data read and write through the MIG7 IP core in the physical layer. The maximum data input rate of the cache can reach 25 Gb/s.

The physical layer contains DDR3 chips and the MIG7 IP core provided by Xilinx.

The clock network, reset network, and monitor are the top-level support modules. The monitor can inform the user of the state of the modules and ensure their health.

The configuration fabric maintains all the configuration information for the cache. For example, the user input interface configuration is shown in Fig. 3 (supplementary materials, Table S1). By configuring this

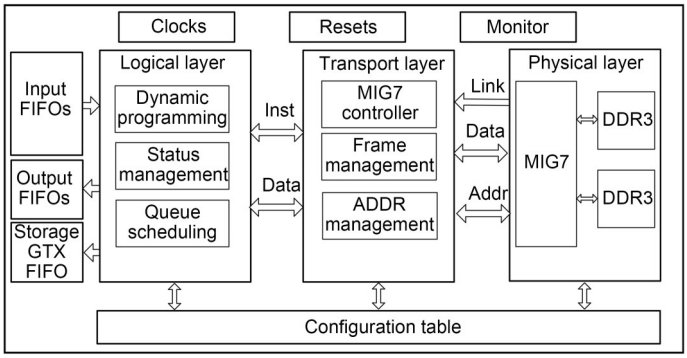

Fig. 2 Hardware architecture of the proposed cache

register, the cache adapts to different numbers of user input interfaces with different features. Users can also customize the usage pattern of the cache controller by configuring others. In other words, the system is open to both users and designers.

3.1.2 Array programmable NAND flash controller

In an $M\times N$ NAND flash array, there are M groups of I/O buses. Each group of NAND flashs is called a cluster, and each cluster of NAND flashs consists of N single dies. To ensure on-orbit reliability, redundant dies are usually included in the design.

Considering the design limitations, such as printed circuit board (PCB) wiring, signal integrity driving capability, and FPGA resources of the microsatellite integrated payload data processing and storage system, a 1×4 NAND flash array is adopted in this paper. It consists of 8×2 NAND flash chips with a total capacity of 2 Tb, and the controller is designed as shown in Fig. 4.

To improve architectural scalability and fault tolerance, the key configurations of the controller, such as data bit width, the number of clusters, and pipeline series, are maintained in the form of a work matrix and work configuration. Users can adjust them for different microsatellites or different mission scenarios. Each partition corresponds to an independent work matrix and work configuration. When several NAND flash dies fail, most of the system functionality can be retained by redefining the work matrix and work configuration without redundant dies. The logical address format, work matrix, and mapping method when one of the dies fails are shown in Fig. 5.

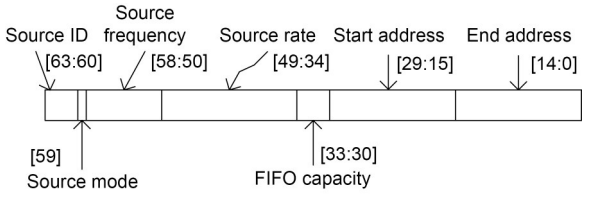

Fig. 3 Arrangement of user input interface configuration

As shown in Fig. 5, the work matrix of the current partition is adjusted to a 4×7 array when the CS6 die fails, and the bit width is set to 32.

Reassigning the partition into two parts for higher performance is another solution during failure, as shown in Fig. 6. When data are written to logical CS00 of partition 0 (bit width set to 32 and other logical CSs of partition 0 set to 64), the programming operations are performed in multiplane mode, with two pages (2 taps) at each operation to align with logical CS01 to CS11 of partition 0. When logical CS00 of partition 0 is full, the remaining halves of logical CS01 to CS11 are partitioned into partition 1 independently. The physical CS4, CS5, and CS6 dies can store data with low write rate requirements, such as TMR storage of payload on-orbit update parameters.

3.2 Performance optimization and analysis

3.2.1 Dynamic programming method

The ADP algorithm can point to the optimal data scheduling policy for payload data input. Nevertheless,

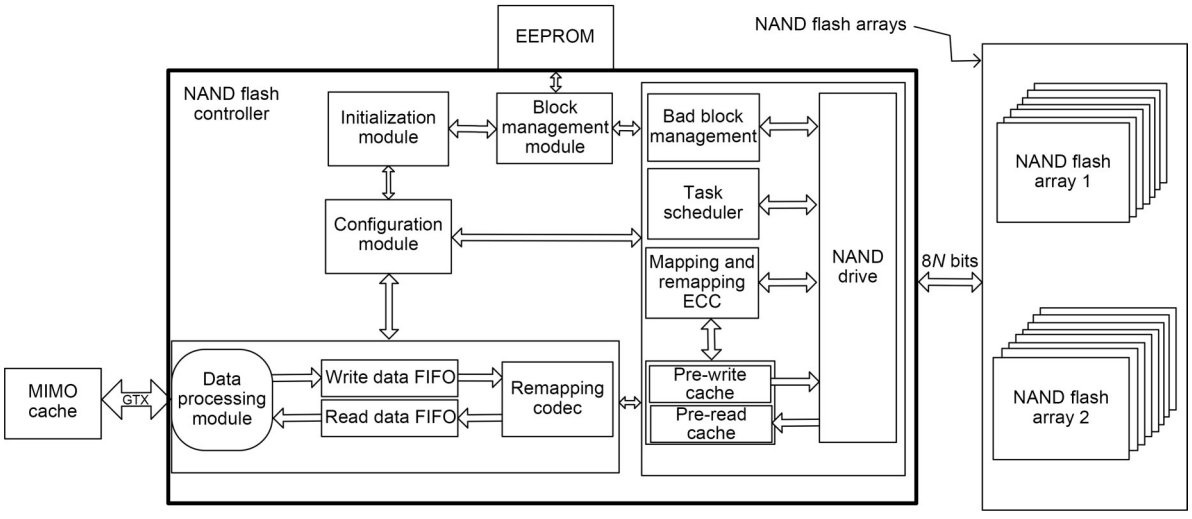

Fig. 4 The hardware architecture of the NAND flash controller

the iterative process of the ADP algorithm incurs considerable FPGA resource consumption, and the code complexity is greatly increased. In this paper, the system autoadjusting weight parameters are replaced by user configurations, and the number of iterations is limited. Fig. 7 shows the design block diagram.

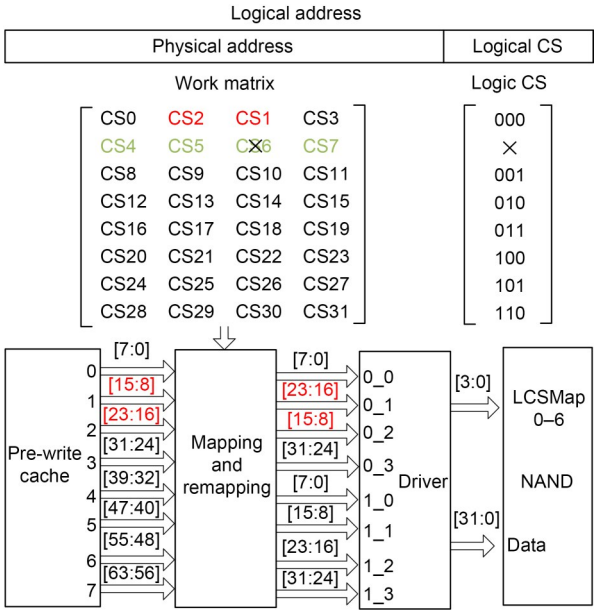

Fig. 5 Data mapping of the NAND flash array

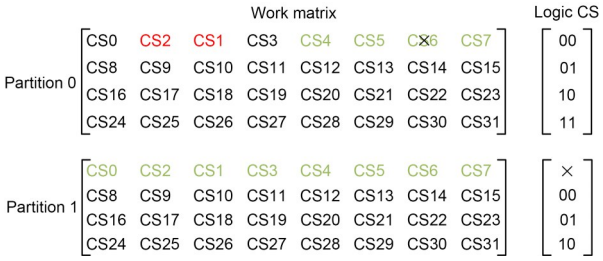

Fig. 6 Example working matrix in the case of failure

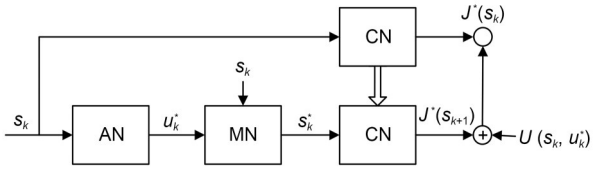

Fig. 7 Structure diagram of dynamic programming

According to Eq. (2), when the write length is less than 1024 bytes, the write efficiency is positively correlated with the write length. In this data exchange problem, the control strategy $u(s_{k,t})$ is to transport $b_{k,t}$ bytes of data into DDR3 from the input FIFO of user k when the state is s_k at time slot t, and $b_{k,t}$ is different due to different user configurations and different control strategies, including dynamic weight scheduling, priority scheduling, polling scheduling, and first-come-first-service (FCFS) scheduling (Li QA, 2019).

The write length is expressed as B_t:

$$B_t = \sum_{k=0}^{K} b_{k,t}. \tag{3}$$

The utility function is expressed as $U(s_{k,t}, u_{k,t})$:

$$U(s_{k,t}, u_{k,t}) = \frac{t_{\mathrm{RCD}} + t_{\mathrm{CWL}} + \frac{B_t}{8} t_{\mathrm{CK}} + t_{\mathrm{WR}} + t_{\mathrm{RP}}}{\frac{B_t}{8} t_{\mathrm{CK}}}, \tag{4}$$
$$0 < B_t \le 1024.$$

Under the control strategy of $u(s_{k,t})$, each input user status can be predicted according to the input user status register (supplementary materials, Table S2). For user k, the remaining time until the FIFO is full can be expressed as $t_{\mathrm{re}}(k, t+1)$ at time slot $t+1$:

$$t_{\mathrm{re}}(k, t+1) = t_{\mathrm{re}}(k, t) + b_{k,t}/v_k - t_{\mathrm{WRITE}}(k, t). \tag{5}$$

The FIFO counter can be expressed as $q(k, t+1)$ at time slot $t+1$:

$$q(k, t+1) = \begin{cases} q(k, t) - b_{k,t}, & t_{\mathrm{re}}(k, t+1) > T_k, \\ q_{\max} - v_k t_{\mathrm{re}}(k, t+1), & t_{\mathrm{re}}(k, t+1) \le T_k, \end{cases} \tag{6}$$

where $t_{\mathrm{WRITE}}(k, t)$ denotes the time required to store B_t bytes of data at time slot t. Based on the FIFO counter and the remaining time at time slot $t+1$, the control strategy $u(s_{k,t+1})$ determines the next state.

Under the control of the control sequence $u_k = \{u(s_{k,t}), u(s_{k,t+1}), u(s_{k,t+2}), \ldots\}$, the performance index function is expressed as $J(s_k, u_k)$:

$$J(s_k, u_k) = \sum_{t=t_0}^{\infty} U(s_{k,t}, u_{k,t}). \tag{7}$$

The optimization objective is to find an admissible control strategy that minimizes the performance index function (7). According to the Bellman optimality principle (Shan and Miura, 2014), the optimal value function satisfies the HJB equation and is expressed as $J^*(s_k)$:

$$J^*(s_k) = \min_{u_k}\{U(s_k, u_k) + J^*(s_{k+1})\} = U(s_k, u_k^*) + J^*(s_{k+1}). \quad (8)$$

To reduce the computational complexity, the number of iterations N is limited, and the performance index function after N iterations is $J_N^*(s_k)$:

$$J_N^*(s_k) = J^*(s_{k+1}) + \sum_{t=0}^{N} U(s_{k,t}, u_{k,t}^*), \quad (9)$$

where u_k^* and $u_{k,t}^*$ denote the optimal control strategies, which are determined by comparing the value of the utility function $U(s_{k,t}, u_{k,t})$ corresponding to different control strategies. Different iteration times N correspond to different scheduling delay times and efficiencies. The optimal N can be obtained by tuning the experimental parameters.

3.2.2 Queue scheduling method

In SASS, the MIMO cache is the access object of the memory module and the real-time processing module. Conventional data distribution strategies, such as distributing data only in priority or request order, cannot meet the requirements of complex tasks with several subtasks.

Combined with the characteristics of real-time systems, a queue scheduling method is proposed to distribute data to each back-end module in this paper. Based on the priority of tasks in the configuration fabric and the status management module, as shown in Fig. 2, the queue scheduling module inserts instructions into the scheduling queue (supplementary materials, Table S3). The instruction execution procedure proceeds as follows:

Step 1: determine whether the instruction format is valid and whether the instruction in the queue is the highest priority instruction.

Step 2: according to the priority parameter of the user output status, the data are outputted simultaneously to the channels in use at the same priority. If priority self-adjusting is enabled, the priority is reevaluated based on the remaining subtasks in the current task and the remaining time $t_{re}(k,t)$ of the user input status.

Step 3: rearrange the instruction queue and execute the next instruction.

3.2.3 NAND flash controller performance analysis

To circumvent the programming time effect on the NAND flash data write rate, a pipelining operation is introduced in SASS, as shown in Fig. 8.

By converting relevant parameters of the MT29F128G08AJAAAWP chip (Micron, 2010), the loading time t_{LOAD} and programming time t_{PROG} can be calculated as follows:

$$t_{PROG} = 350\text{–}560\ \mu s, \quad (10)$$

$$t_{LOAD} = (n_C + n_A + n_D) \times t_{WC} + t_{WB} + t_{ADL} = (2 + 5 + 8640) \times 25\ ns + 100\ ns + 70\ ns = 216\,345\ ns, \quad (11)$$

where t_{WC} is the write cycle, which is at least 20 ns in the chip manual, and 25 ns is adopted in SASS for better signal quality and lower BER. t_{WB} is the interval from the end of data loading to the selected die going busy for t_{PROG}, and t_{ADL} is the interval from the end of the command to the start of data. Four levels of pipelining are sufficient for seamless data loading. The maximum write rate of the controller is expressed as v_{PROG_MAX}:

$$v_{PROG\_MAX} = \frac{N_p \times N_d \times 64}{t_{PROG} + N_p \times t_{LOAD}} \approx 2555.92\ \text{Mb/s}, \quad (12)$$

where N_p is the number of pages and N_d is the number of bytes of data per page.

When a partition is fully written and enters the cyclic mode, the controller automatically erases the oldest data after each storage task to ensure that the erase pointer precedes the write pointer, as shown in Fig. 9.

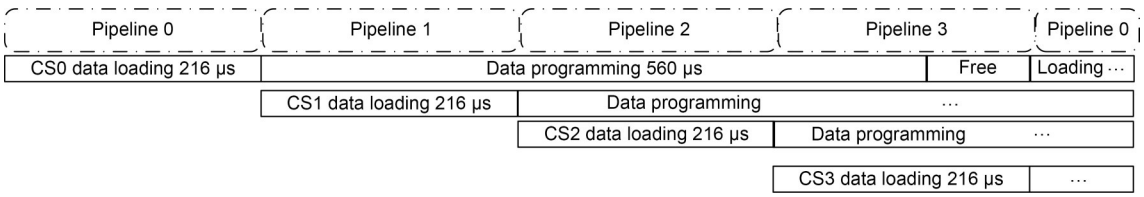

Fig. 8 NAND flash controller pipeline operation sequence diagram

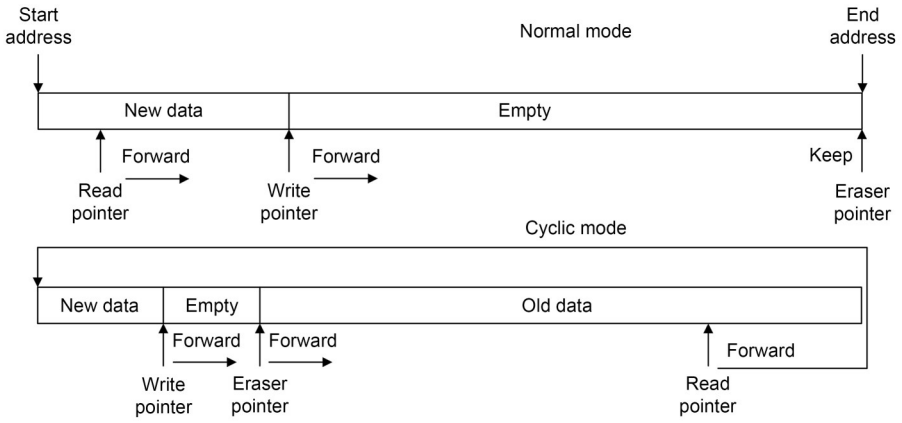

Fig. 9 Cyclic mode of the NAND flash controller

The amount of data to be erased is determined by the user configurations. When the NAND flash controller is idle, the user can manipulate it with only data and simple commands. In addition, when the erase instruction is received from the host computer, the write pointer remains in place and the empty blocks are not erased repeatedly to ensure P/E balance of each block.

The write rate in the case of Fig. 5 is expressed as $v_{\mathrm{PROG\_4\times7}}$:

$$v_{\mathrm{PROG\_4\times7}}=\frac{N_{\mathrm{p}}\times N_{\mathrm{d}}\times 32}{t_{\mathrm{PROG}}+N_{\mathrm{p}}\times t_{\mathrm{LOAD}}}\approx 1277.96\ \mathrm{Mb/s}. \quad (13)$$

The write rate in the case of Fig. 6 is expressed as $v_{\mathrm{PROG\_P0}}$ and $v_{\mathrm{PROG\_P1}}$:

$$v_{\mathrm{PROG\_P0}}=\frac{N_{\mathrm{p}}\times N_{\mathrm{d}}\times 64}{t_{\mathrm{PROG}}+N_{\mathrm{p}}/4\times(5\times t_{\mathrm{LOAD}}+t_{\mathrm{DBSY}})} \quad (14)$$
$$\approx 2042.846\ \mathrm{Mb/s},$$
$$v_{\mathrm{PROG\_P1}}=\frac{N_{\mathrm{p}}\times N_{\mathrm{d}}\times 64}{t_{\mathrm{PROG}}+N_{\mathrm{p}}/3\times(t_{\mathrm{LOAD}}+t_{\mathrm{PROG}})} \quad (15)$$
$$\approx 2137.731\ \mathrm{Mb/s},$$

where t_{DBSY} is the busy time after loading the first plane. As described above, the NAND flash controller can retain most of its performance even in the presence of the storage media failure.

3.3 Data reliability scheme

3.3.1 Zero-delay configurable error correction scheme

In this paper, an adjustable Hamming ECC module with zero delays is designed to correct bit errors and avoid the effects of the codec process on read and write rates. The ECC module performs row checks and column checksums on $M\times N$ ($2^m\times 2^n$, $0\le m\le 3$, $3\le n\le 9$) bytes of data each time, and performs XOR calculations on each bit to be checked, as shown in Fig. 10.

For $2^m\times 2^n$ bytes of data, the row and column checksums are denoted as RP and CP respectively, as shown in Fig. 11.

$$\begin{cases}\mathrm{RP}_{2k}=\text{xor bit0 to bit}2^{m+3},\ \mathrm{RowCounter}(k)=0,\\ \mathrm{RP}_{2k+1}=\text{xor bit0 to bit}2^{m+3},\ \mathrm{RowCounter}(k)=1,\\ \mathrm{CP}_{2t}=\text{xor bit}i,\ i(t)=0,\\ \mathrm{CP}_{2t+1}=\text{xor bit}i,\ i(t)=1,\end{cases} \quad (16)$$

where RowCounter(k) means the k^{th} bit of row counter and $i(t)$ means the i^{th} bit in Fig. 11.

The length of the ECC for each check segment is $2(m+n+3)$ bits, and the maximum length per data page is 384 bytes. By varying m, the user can choose to manage individual NAND flash chips independently or to manage the entire cluster uniformly. When a small amount of noise in the remote sensing data can be tolerated and higher storage efficiency is needed, n can be adjusted to increase the length of each check segment and then decrease the total length $\mathrm{cnt}_{\mathrm{oob}}$ of the ECC. The storage efficiency is expressed as η_{WRITE}:

$$\eta_{\mathrm{WRITE}}=\frac{8192}{8192+\mathrm{cnt}_{\mathrm{oob}}}. \quad (17)$$

The process of the Hamming ECC module is provided in Table S4 in the supplementary materials.

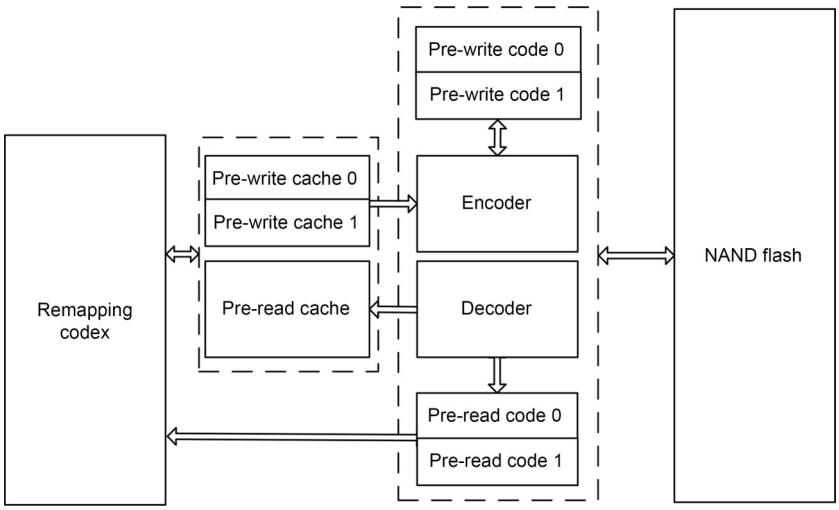

Fig. 10 Architecture of the Hamming error correction code module

Row counter	Bit*i*												
Byte0	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	RP0	RP2	RP4	…	RP16
Byte1	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	RP1				
Byte2	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	RP0	RP3			
Byte3	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	RP1				
…	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0					
Byte508	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	RP0	RP2	RP5		RP17
Byte509	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	RP1				
Byte510	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	RP0	RP3			
Byte511	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	RP1				
	CP1	CP0	CP1	CP0	CP1	CP0	CP1	CP0					
	CP3		CP2		CP3		CP2						
	CP6				CP5								

Fig. 11 Schematic of the error correction code

3.3.2 Data remapping method

As analyzed in Section 2.3, the service life of NAND flash can be extended by reducing the data BER. This paper proposes a data remapping method to achieve this and reduce the number of P/E operations.

In view of the characteristics of NAND flash, reducing the number of negative electronic operations means an increase in the ratio of "1" state in memory cells, which is beneficial for decreasing both retention errors and program disturbance errors accordingly. It can also reduce the number of P/E operations. This increase can be achieved by increasing the ratio of "1" in the programming data.

The remapping method is implemented by counting the numbers (matrix $\boldsymbol{N}$) of 00, 01, 10, and 11 (matrix $\boldsymbol{M}_0=[00, 01, 10, 11]^{\mathrm{T}}$) when prestoring data into the prewrite cache as shown in Fig. 4 and remapping $\boldsymbol{M}_0$ to matrix $\boldsymbol{M}_{\mathrm{optimal}}$ with the largest gain $G_{\max}$ for "1" among its permutations, which is expressed as $\boldsymbol{M}_{\mathrm{p}}$:

$$\begin{aligned}\boldsymbol{M}_{\mathrm{p}} &= \left[\boldsymbol{M}_1, \boldsymbol{M}_2, ..., \boldsymbol{M}_{23}\right] \\ &= \begin{bmatrix} 00\ 00\ 00\ 00\ 00\ 01\ ... \\ 01\ 10\ 10\ 11\ 11\ 00\ ... \\ 11\ 01\ 11\ 01\ 10\ 10\ ... \\ 10\ 11\ 01\ 10\ 01\ 11\ ... \end{bmatrix}. \end{aligned} \tag{18}$$

The gain of remapping $\boldsymbol{M}_0$ to $\boldsymbol{M}_n$ is $\boldsymbol{g}_n$ (n=1, 2, ..., 23), and the gain matrix is expressed as $\boldsymbol{g}_{\mathrm{p}}$ after removing the repetition matrix and matrix $\boldsymbol{O}$. The total gain after remapping is $\boldsymbol{G}$, and the maximum value from G_1 to G_{11} is $G_{\max}$.

$$\boldsymbol{g}_{\mathrm{p}}=\left[\boldsymbol{g}_1,\boldsymbol{g}_2,\ldots,\boldsymbol{g}_{11}\right]=\begin{bmatrix}0&0&1&1&1&1&1&1&2&2&2\\0&1&-1&-1&0&0&1&1&-1&0&0\\1&0&0&1&-1&1&-1&0&0&-1&0\\-1&-1&0&-1&0&-2&-1&-2&-1&-1&-2\end{bmatrix},\quad(19)$$

$$\boldsymbol{G}=\boldsymbol{N}\times\boldsymbol{g}_{\mathrm{p}}=[G_1,G_2,\ldots,G_{11}],\quad(20)$$

$$G_{\max}=\max\{G_1,G_2,\ldots,G_{11}\},\quad(21)$$

$$\boldsymbol{M}_{\mathrm{optimal}}=\boldsymbol{M}_n,\ \text{if}\ G_{\max}=G_n.\quad(22)$$

Taking 8192 bytes of data in Table 1 as an example, the ratio of 1 in the data increases from 44.94% to 68.25% in No. 1, from 68.28% to 72.47% in No. 2, and from 21.35% to 78.65% in No. 3. Before programming, the data are encoded according to the remapping rule of $\boldsymbol{M}_0$ to $\boldsymbol{M}_{\mathrm{optimal}}$, and $\boldsymbol{M}_{\mathrm{optimal}}$ is stored in the out-of-band (OOB) section of each page of NAND flash, as shown in Fig. 12.

Table 1 Example of data remapping

No.	$\boldsymbol{M}_0$	$\boldsymbol{N}$	$G_{\max}$	$\boldsymbol{M}_{\mathrm{optimal}}$
1	00	7160	G_6, 15274	10
	01	2209		01
	10	18304		11
	11	5095		00
2	00	6198	G_5, 2741	10
	01	4038		01
	10	3457		00
	11	19075		11
3	00	21319	G_{11}, 37550	11
	01	4919		01
	10	3986		10
	11	2544		00

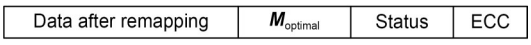

Fig. 12 Storage of $\boldsymbol{M}_{\mathrm{optimal}}$ on the NAND flash page

When data are read, the remapped data are stored in the preread cache, and data error correction and decoding are performed in sequence.

3.3.3 Block management scheme

By analyzing the characteristics of the remote sensing payload data, a block management module implements the wear balancing, file system, and bad block management. In Table 2, the packet size is denoted by V_k, F_k designates the packet frequency, and the residence time R_k has three levels.

Table 2 Characteristics of the main remote sensing payload data

Type	Packet size	Packet frequency	Residence time level
Number of image packets	Huge	High	1
Number of video packets	Huge	High	1
Number of target packets	Medium	Medium	1
Processing results	Little	Low	2
Telemetry data	Little	Low	2
Number of on-orbit update packets	Little	Low	3

Based on the characteristics of the data, the NAND flash arrays are partitioned to ensure the wear balancing of each block and to meet the reliability requirements of special types of data, such as on-orbit update packets, including codes and parameter libraries for functional reconstruction of the payload system. The partition size N_k is expressed as follows:

$$N_k=N_{\mathrm{total}}\frac{V_kF_kR_k}{\sum_{n=0}^{N}V_nF_nR_n}.\quad(23)$$

In remote sensing satellites, payload work is generally coordinated by the mission planning system. A file index table (FIT) is built in SASS for easy retrieval of data and consists of mission ID, packet ID or version ID, timestamps, and block addresses to replace the cumbersome file system, as shown in Fig. 13.

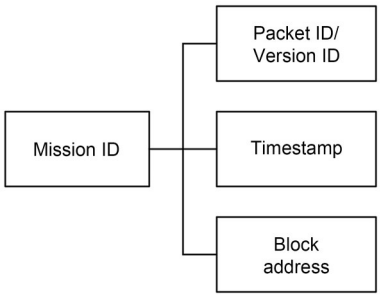

Fig. 13 Composition of the file index table

FIT is stored in three electrically erasable programmable read-only memory (EEPROM) chips in TMR mode to ensure data reliability, so are the block state bitmap, the bad block mapping table, the operation pointers, and the weakly-performed page address, as shown in Fig. 14.

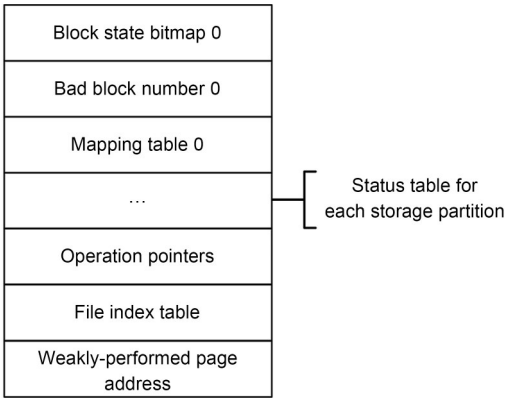

Fig. 14 Data storage structure in EEPROM

The host records the address of a page whose error correction fails for multiple segments. When the number of error correction failures on this page exceeds the threshold, the host marks it as a weakly-performed page in Fig. 14 and it will be skipped in future operations.

Common approaches for bad block management are the skip-block and reserve-block methods. However, the skip-block method has difficulty in achieving transparent bad block management and has poor portability. This paper proposes a reverse-order retrieval method based on the reserve-block method. The bad block table retrieval flowchart and the bad block table establishment flowchart are shown in Figs. 15 and 16, respectively.

A bad block in a reservation needs to be skipped and marked in a regular design. Based on this paper's reverse-order retrieval method, another address is mapped to the original bad block. Then, when retrieving the mapping table in reverse order, the first corresponding address retrieved is the valid address.

Overall, the block management module simplifies and integrates the file system, the wear balancing algorithm, and the bad block management algorithm concisely and efficiently, and meets the storage and playback requirements of remote sensing payload data in microsatellites.

4 Experiments and results

4.1 Experimental settings

To validate the excellence of SASS, an experimental platform was designed and implemented, and SASS was deployed to the hardware, as shown in Fig. 17.

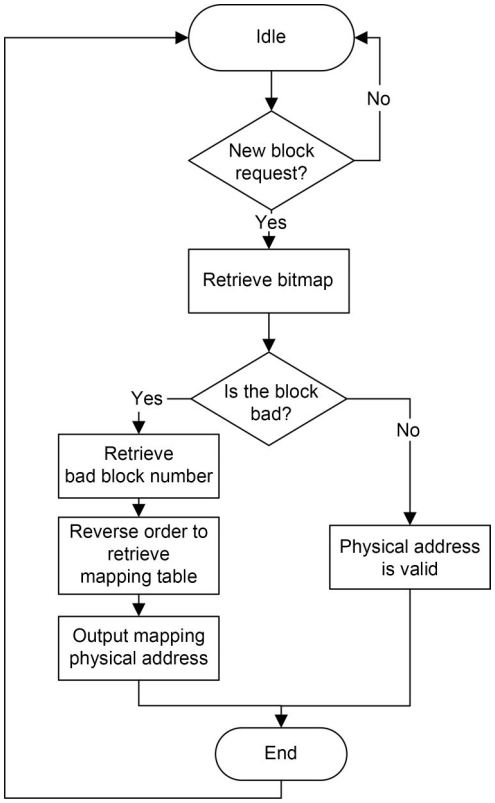

Fig. 15 Flowchart of bad block table retrieval

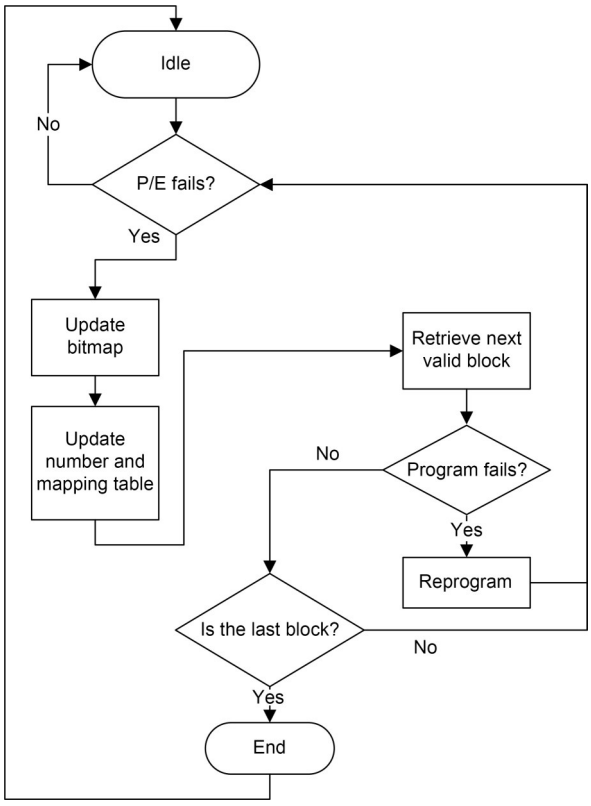

Fig. 16 Flowchart of bad block table establishment

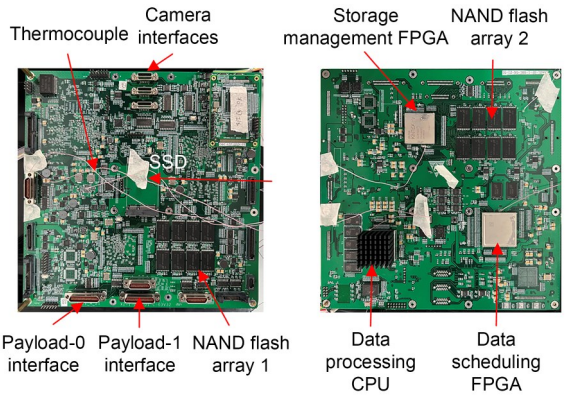

Fig. 17 Photographs of the experimental platform

The platform was designed based on COTS devices, with a weight of 331 g, a volume of 222 m× 222 m×30.7 mm, and a maximum power consumption of 30 W. The minimum power consumption is 3.7 W when the platform works only for payload management and data down transmission.

To verify the scalability of SASS, this work assembled the experimental platform in different models of microsatellites (supplementary materials, Table S5) for the thermal balance test, thermal vacuum test, thermal cycling test, radiation test, electromagnetic compatibility test, and desktop aging test. The system worked properly in each test. Moreover, extensive experiments were conducted to evaluate the performance and reliability of SASS.

4.2 Performance evaluation of the MIMO cache

Based on a model *A* microsatellite, the camera-1 data scheduling task was set to the highest priority. Then we triggered the image acquisition of camera 1 with instruction, and scheduled data with dynamic weight (DW) scheduling, priority scheduling, polling scheduling, FCFS scheduling, and dynamic programming (DP) methods. Then, the time interval and measured DDR3 access bandwidth from the beginning of camera-1 data collection to the end of distribution were recorded to evaluate the performance of the DP method and the queue scheduling method. The results are shown in Table 3 and the figure is provided in Fig. S1 in the supplementary materials.

Table 3 shows that the methods can effectively reduce the data scheduling time in multipayload microsatellites, with N=2 being a more appropriate configuration in this experimental scenario.

Table 3 Scheduling time of camera-1 data

Method	Time (ms)
Dynamic weight	287.1
Priority	167.9
Polling	354.6
FCFS	205.2
DP1	164.2
DP2	163.5
DP4	165.9

FCFS: first-come-first-service; DP: dynamic programming

4.3 Performance evaluation of the NAND flash array and controller

The performance of the nonpipelined 4×8 mode (64 bits), 4-level pipelined 4×8 mode (64 bits), 8-level pipelined 8×4 mode (32 bits), and fault mode as shown in Fig. 6 was evaluated. In different modes, 5A codes (bytes all 0x55 or 0xAA) and images were written into a partition of 256 Gb approximately 570 images and the average write rate was recorded. These images were taken during ground experiments with a spaceborne high-resolution camera (supplementary materials, Fig. S2). The results are in Figs. 18 and 19.

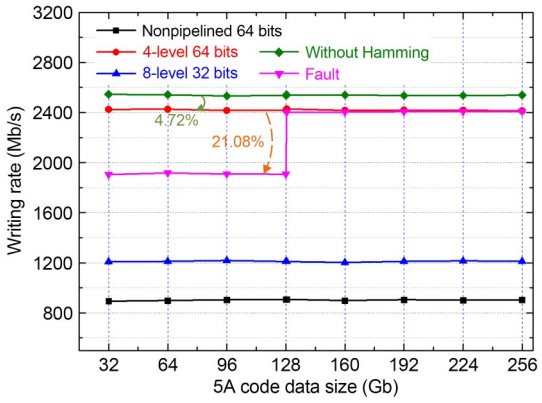

Fig. 18 Write rate of 5A codes

Figs. 18 and 19 show that the maximum storage rates of the 4-level pipelined 4×8 mode (64 bits) and 8-level pipelined 8×4 mode (32 bits) were 2429 and 1212 Mb/s, respectively, which were almost the same in the two figures. The rate reduction due to the Hamming code in the OOB section was 4.72% in Fig. 18 and 4.91% in Fig. 19. The matrix $\boldsymbol{M}_{\text{optimal}}$ took up only 3 bytes, which reduced the storage efficiency very little.

For the nonpipelined mode, the rate of the 5A code experiment was significantly higher and more

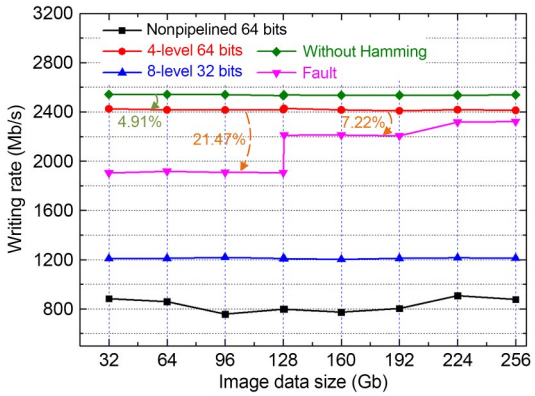

Fig. 19 Write rate of images

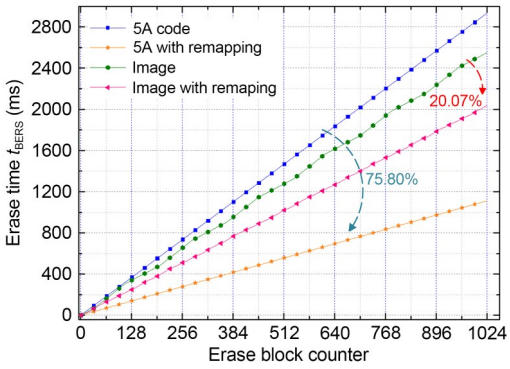

Fig. 20 Erase block operation time t_{BERS}

stable than the rate of the imaging experiment, and the same went for the failure mode.

In Fig. 18, the 5A code was remapped to the 0xFF code, so the rate in the second half of the fault mode was basically the same as that in the 4-level pipelined 4×8 mode (64 bits), which means that the 3-level pipelined 3×8 mode (64 bits) can also meet the seamless data loading requirement under this experimental condition. In Fig. 19, the write rate jitter of the fault mode depended on the difference in the ratio of 1 in the image after remapping, and the performance of the first and second halves of the fault mode was preserved by 78.53% and 92.78%, respectively.

The higher the ratio of "1," the shorter the time of the write operation and the erase operation. The pipelined design masked the effect of the program page operation time t_{PROG}, but the effect of the erase block operation time t_{BERS} remained. This experiment also evaluated the performance of the remapping method on the NAND flash operations, and the results are shown below. In Fig. 20, t_{BERS} of the 5A code was reduced by 75.80% and t_{BERS} of the remapped image was reduced by 20.07%.

4.4 Reliability evaluation

Due to the data remapping method, more NAND cells remain in the lowest voltage level (voltage level of "1") and the retention and program disturbance errors of NAND flash are reduced. The Hamming error correction module can correct a one-bit error in each check segment (512 bytes/1 bit for images). To verify the effectiveness of the methods, experiments were performed according to the following experimental matrix in Table 4.

Table 4 Experimental matrix

No.	Remapping	Hamming	Temperature cycling
1	×	×	×
2	√	×	×
3	×	√	×
4	√	√	×
5	×	×	√
6	√	×	√
7	×	√	√
8	√	√	√

Under each experimental condition, the data P/E cycle experiment and the data retention experiment were performed. To compare BER differences more clearly, 30 000 P/E cycles were applied to the experimental data blocks in advance, and the maximum number of the recommended P/E cycles for this block was 60 000.

According to the thermal control requirements of the system in the satellite, the experiments were carried out under the thermal cycling condition of −30 °C to 60 °C (the chip temperature ranged from −17 °C to 79 °C), and the maximum temperature change was 5 °C/min. The holding time at high and low temperatures was 2 h. During the holding time, the data were written to the system, and the BER was collected.

The results of the experiments are shown in Table 5, and the figures are provided in Figs. S3–S7 in the supplementary materials.

According to the experimental results, the proposed method can significantly reduce program disturbance and data retention errors at room temperature and during temperature cycling. It was also effective when Hamming code error correction was enabled. It is worth noting that at 60 °C and 64 000 P/E cycles

Table 5 Performance of the proposed method

Condition		Error reduction (%)		
		Only remapping	Only Hamming	Both
Program disturbance	Room	49.35	79.78	95.86
	−30 ℃	42.35	87.20	96.82
	60 ℃	37.80	73.39	89.47
Retention	Room	50.73	92.94	99.16
	Cycle	52.80	90.09	98.31

Room: room temperature; cycle: temperature cycling

(BER>3.55×10^{-4}), the disturbance BER exceeded the error correction ability of the Hamming code, and the remapping method delayed this threshold to 74 000 P/E cycles (supplementary materials, Fig. S4).

4.5 Power evaluation

The experimental platform consists of two FPGAs and an image processing DSP. SASS is only part of it. When the DSP was in the standby state, the power consumption of the platform of different modules deleted was recorded in Table 6. The results showed that the total power consumption of the SASS was approximately 8.5 W, much smaller than that of the Calculex CSR-6602. The proposed method increased the system power consumption, which was relatively small.

Table 6 Power consumption under different conditions

Condition	Power consumption (W)		
	−30 ℃	Room	60 ℃
Only NAND flash controller	1.370	1.397	1.720
Only MIMO cache	6.490	7.101	8.700
Without DP method	16.910	18.362	22.100
Without queue scheduling	17.250	18.727	22.477
Without Hamming codex	17.373	18.893	22.643
Without data remapping	17.360	18.870	22.628
All functions	17.384	18.901	22.650

Room: room temperature

5 Conclusions

In this paper, SASS is proposed and implemented with COTS FPGAs and memory devices for remote sensing microsatellites to cope with increasingly complex remote sensing payloads and real-time processing scenarios in microsatellites.

The system has excellent scalability under modular, standardized, low-coupling, and controllable architecture designs. Small functional particles improve the robustness of the system.

The maximum instantaneous data throughput rate of the MIMO cache of the storage system can reach 25 Gb/s. The NAND flash array and controller can provide an average storage rate of up to 2429 Mb/s (a total of 256 Gb, approximately 570 images). When one of the 32 NAND flash dies fails, at least 78.53% and up to 92.78% of the performance can be preserved, which still meets the image acquisition requirements of spaceborne high-resolution cameras (1717 Mb/s). The proposed dynamic programming method and queue scheduling method effectively improve the efficiency of data scheduling.

The data remapping method can reduce the retention error by at least 50.73% and the program disturbance error by at least 37.80%. It makes up for the limitation of the error correction algorithm to some extent. Combined with the zero-delay configurable error correction scheme, the reduction can be up to 96.82% in program disturbance BER and 99.16% in retention BER (totally 256 Gb, approximately 570 images). With the block management module, users and host computers can manage the storage system transparently and conveniently, which makes the storage system more reliable and flexible.

In addition, as a subsystem of a microsatellite, SASS meets the requirements for microsatellite limited SWaP and has been verified by microsatellite reliability tests and engineering tests as described in Section 4.1.

In the future, we will further study the data characteristics of remote sensing satellites and optimize the scheduling algorithm of the MIMO cache to obtain a better dynamic programming effect, especially to build a mathematical model or formula to calculate the optimal number of iterations N by analyzing the ADP algorithms and data characteristics rather than through testing and tuning. On the other hand, more integrated storage media, such as MLC, TLC NAND flash, and solid state drive (SSD), will be considered for use in SASS (Zhu et al., 2020), combined with more efficient coding methods to meet the increasing storage capacity, write rate, and reliability requirements.

Contributors

Shilei TU and Huiquan WANG designed the research. Shilei TU and Yue HUANG processed the data. Shilei TU drafted the paper. Huiquan WANG and Zhonghe JIN offered advice. Shilei TU, Huiquan WANG, and Zhonghe JIN revised and finalized the paper.

Conflict of interest

Zhonghe JIN is an editorial board member of *Frontiers of Information Technology & Electronic Engineering*, and he was not involved with the peer review process of this paper. All the authors declare that they have no conflict of interest.

Data availability

The data that support the findings of this study are available from the corresponding author upon reasonable request.

References

Airbus Defence and Space, 2022. Airbus Defence and Space S.A.S. http://www.astrium.eads.net/ [Accessed on Sept. 23, 2022].

Cai Y, Mutlu O, Haratsch EF, et al., 2013. Program interference in MLC NAND flash memory: characterization, modeling, and mitigation. Proc IEEE 31st Int Conf on Computer Design, p.123-130. https://doi.org/10.1109/ICCD.2013.6657034

Calculex, 2022. Calculex Electronics, Systems, Software and Support. https://www.calculex.com/A_home.html [Accessed on Sept. 23, 2022].

Ding WZ, Li XH, Yang H, 2020. Quick-response microsatellite constellation design. Proc IEEE 4th Information Technology, Networking, Electronic and Automation Control Conf, p.262-267. https://doi.org/10.1109/ITNEC48623.2020.9084884

Dong ZX, 2017. Research on Application of Spaceborne Solid State Recorder File Management Scheme. PhD Thesis, National Space Science Center, University of Chinese Academy of Sciences, Beijing, China (in Chinese).

Fabiano M, Furano G, 2013. NAND flash storage technology for mission-critical space applications. *IEEE Aerosp Electron Syst Mag*, 28(9):30-36. https://doi.org/10.1109/MAES.2013.6617096

Huang XP, Netravali R, Man H, et al., 2012. Multi-sensor fusion of electro-optic and infrared signals for high resolution visible images: part II. IEEE OCEANS, p.1-6. https://doi.org/10.1109/OCEANS.2012.6405013

Kong LH, He W, Yang CG, et al., 2021. Robust neurooptimal control for a robot via adaptive dynamic programming. *IEEE Trans Neur Netw Learn Syst*, 32(6):2584-2594. https://doi.org/10.1109/TNNLS.2020.3006850

Li QA, 2019. Virtual Multichannel Storage Technology for Large Capacity Spaceborne Data. MS Thesis, Huazhong University of Science & Technology, Wuhan, China (in Chinese). https://doi.org/10.27157/d.cnki.ghzku.2019.004122

Li S, 2015. Study on the Space-borne Solid State Recorder—about Hardware Architecture. PhD Thesis, Center for Space Science and Applied Research, University of Chinese Academy of Sciences, Beijing, China (in Chinese).

Liu GH, Zhou J, Sun F, 2018. Design and in-orbit verification of data storage system for nano-pico satellites. *J Harbin Inst Technol*, 50(2):178-183 (in Chinese). https://doi.org/10.11918/j.issn.0367-6234.201706176

Liu ZX, 2021. Research on Integrated Avionics for Remote Sensing Satellite. PhD Thesis, University of Science and Technology of China, Hefei, China (in Chinese). https://doi.org/10.27517/d.cnki.gzkju.2021.000782

Mahdy AM, Abdelaziz M, El-Azeem MHA, 2020. Design and simulation of parallel BCH code with LDPC code for flash memories. Proc 12th Int Conf on Electrical Engineering, p.196-199. https://doi.org/10.1109/ICEENG45378.2020.9171743

Micron, 2010. Micron NAND Flash Memory. https://www.semiee.com/file2/165a2193073703c940800196ff67e021/Micron/Micron-MT29F%20series.pdf [Accessed on Sept. 23, 2022].

Micron, 2021. MT41K512M16HA-125. https://www.semiee.com/file2/dabbd658a6c01cbc1846c4bc799bf821/Micron/Micron-MT41K512M16HA-125.pdf [Accessed on Sept. 23, 2022].

Pavan P, Bez R, Olivo P, et al., 1997. Flash memory cells—an overview. *Proc IEEE*, 85(8):1248-1271. https://doi.org/10.1109/5.622505

Radhakrishnan R, Edmonson WW, Afghah F, et al., 2016. Survey of inter-satellite communication for small satellite systems: physical layer to network layer view. *IEEE Commun Surv Tutor*, 18(4):2442-2473. https://doi.org/10.1109/COMST.2016.2564990

Reid M, Ottman G, 2014. Software controlled memory scrubbing for the Van Allen Probes solid state recorder (SSR) memory. IEEE Aerospace Conf, p.1-6. https://doi.org/10.1109/AERO.2014.6836406

Rhee S, Kim C, Kim J, et al., 2010. Concatenated Reed–Solomon code with hamming code for DRAM controller. Proc 2nd Int Conf on Computer Engineering and Applications, p.291-295. https://doi.org/10.1109/ICCEA.2010.65

Sasada T, Ichikawa S, Shirakura M, 2005. Mass data recorder with ultra-high-density stacked memory for spacecraft. IEEE Aerospace Conf, p.1-8. https://doi.org/10.1109/AERO.2005.1559545

Schmidt AG, French M, Flatley T, 2017. Radiation hardening by software techniques on FPGAs: flight experiment evaluation and results. IEEE Aerospace Conf, p.1-8. https://doi.org/10.1109/AERO.2017.7943651

SEAKR Engineering, 2022. Aerospace Data Storage and Processing Solutions. http://www.seakr.com/ [Accessed on Sept. 23, 2022].

Sezer A, Çelikkaya O, Hassoy B, 2017. Microsatellite constellation for Earth observation missions. Proc 8th Int Conf on Recent Advances in Space Technologies, p.279-284. https://doi.org/10.1109/RAST.2017.8002944

Shan L, Miura R, 2014. Energy-efficient scheduling under hard delay constraints for multi-user MIMO system. Int Symp on Wireless Personal Multimedia Communications, p.696-699. https://doi.org/10.1109/WPMC.2014.7014905

Song Q, 2015. Research and Application of Storage Management Technology for Spaceborne Solid State Recorder. PhD Thesis, Center for Space Science and Applied Research,

University of Chinese Academy of Sciences, Beijing, China (in Chinese).

Space Micro, 2022. Space Micro. http://www.spacemicro.com [Accessed on Sept. 23, 2022].

Stuffler T, Kaufmann C, Hofer S, et al., 2007. The EnMAP hyperspectral imager—an advanced optical payload for future applications in Earth observation programmes. *Acta Astronaut*, 61(1-6):115-120. https://doi.org/10.1016/j.actaastro.2007.01.033

Vakil A, Blasch E, Ewing R, et al., 2021. Visualizations of fusion of electro optical (EO) and passive radio-frequency (PRF) data. IEEE National Aerospace and Electronics Conf, p.294-301. https://doi.org/10.1109/NAECON49338.2021.9696424

Wang GQ, Chen H, Xie YZ, 2021. An efficient dual-channel data storage and access method for spaceborne synthetic aperture radar real-time processing. *Electronics*, 10(6):662. https://doi.org/10.3390/ELECTRONICS10060662

Wang JX, 2010. Research and Implementation on Solid State Storage Technology Based on NAND Flash Memory. MS Thesis, National University of Defense Technology, Changsha, China (in Chinese).

Wang LG, 2019. Research on High Performance Integrated Aerospace Electronic Technology. PhD Thesis, National Space Science Center, University of Chinese Academy of Sciences, Beijing, China (in Chinese).

Wei DB, Deng LB, Qiao LY, et al., 2016. PEVA: a page endurance variance aware strategy for the lifetime extension of NAND flash. *IEEE Trans Very Large Scale Integr (VLSI) Syst*, 24(5):1749-1760. https://doi.org/10.1109/TVLSI.2015.2479250

Werbos PJ, 2011. Computational intelligence for the smart grid-history, challenges, and opportunities. *IEEE Comput Intell Mag*, 6(3):14-21. https://doi.org/10.1109/MCI.2011.941587

Xiao YA, Luo CL, Yang C, 2011. The comparative analysis of LDPC and RS code. Int Conf on Consumer Electronics, Communications and Networks, p.4510-4513. https://doi.org/10.1109/CECNET.2011.5768345

Xu W, Jin G, Wang JQ, 2017. Optical imaging technology of JL-1 lightweight high resolution multispectral remote sensing satellite. *Opt Prec Eng*, 25(8):1969-1978 (in Chinese). https://doi.org/10.3788/OPE.20172508.1969

Xu X, Hu DW, Lu XC, 2007. Kernel-based least squares policy iteration for reinforcement learning. *IEEE Trans Neur Netw*, 18(4):973-992. https://doi.org/10.1109/TNN.2007.899161

Xu X, Yang HY, Lian CQ, et al., 2017. Self-learning control using dual heuristic programming with global Laplacian Eigenmaps. *IEEE Trans Ind Electron*, 64(12):9517-9526. https://doi.org/10.1109/TIE.2017.2708002

Yang CL, 2008. Research and Implementation of Embedded File System Based on NAND Flash. MS Thesis, University of Electronic Science and Technology of China, Chengdu, China (in Chinese).

Yang Z, Long T, 2015. Methods to improve system verification efficiency in FPGA-based spaceborne SAR image processing system. IET Int Radar Conf, p.1-5. https://doi.org/10.1049/cp.2015.1310

Yin YN, 2018. Investigation of Single Event Effects and Synergistic Effects with Total Ionizing Dose in Flash Memory. PhD Thesis, Institute of Modern Physics, University of Chinese Academy of Sciences, Beijing, China (in Chinese).

Zhang XY, 2020. Research of High Speed Synchronous NAND Flash Spaceborne Storage Technology. MS Thesis, National Space Science Center, University of Chinese Academy of Sciences, Beijing, China (in Chinese). https://doi.org/10.27562/d.cnki.gkyyz.2020.000070

Zhang Y, Zhou Q, Li J, et al., 2015. The simulation and realization of input-buffer scheduling algorithm in satellite switching system. *Space Electron Technol*, 12(1):97-103. https://doi.org/10.3969/j.issn.1674-7135.2015.01.020

Zhang Y, Xia Z, Zhang T, et al., 2020. Real-time processing of ship detection with spaceborne SAR image based on multiple FPGA. IET Int Radar Conf, p.1060-1065. https://doi.org/10.1049/icp.2021.0684

Zhou F, Yang J, Sun GC, et al., 2020. A real-time imaging processing method based on modified RMA with sub-aperture images fusion for spaceborne spotlight SAR. IEEE Int Geoscience and Remote Sensing Symp, p.1905-1908. https://doi.org/10.1109/IGARSS39084.2020.9324118

Zhu RZ, Tang L, Wan J, et al., 2020. Development trend of spaceborne storage system. *Microelectron Comput*, 37(12):59-63. https://doi.org/10.19304/j.cnki.issn1000-7180.2020.12.012

List of supplementary materials

Table S1 Bit definitions of user input interface configuration

Table S2 Bit definitions of user input status

Table S3 Bit definitions of queue scheduling instruction

Table S4 Hamming code codec process

Table S5 Payload matrix and parameters of different microsatellites

Fig. S1 Measured DDR3 access bandwidth

Fig. S2 Images used in the experiments

Fig. S3 Program disturbance BER at the room temperature

Fig. S4 Program disturbance BER at −30 ℃

Fig. S5 Program disturbance BER at 60 ℃

Fig. S6 Data retention BER at the room temperature

Fig. S7 Data retention BER during temperature cycling