Embedded Software Development for MSC on KOMPSAT-2

H.P.Heo, J.P.Kong, S.S.Yong, Y.S.Kim, J.E.Park, H.S.Youn, H.Y.Paik
Korea Aerospace Research Institute
45 Eun-dong Yusung-gu, Taejon 305-600, Korea
(hpyoung, kjp123, ssyong, yskim1203, pje, youn, phy) @kari.re.kr

Abstract: MSC(Multi-Spectral Camera) system is a remote sensing instrument to obtain high resolution ground image. MSC system includes main control unit, called SBC(Single Board Computer). SBC controls all the sub-units of MSC system and communicates with spacecraft bus. The software developed for SBC should be reliable and autonomous to support various kinds of imaging missions. It is being developed using VxWorks real-time operating system to manage all tasks for all units efficiently. In this paper, the characteristics of the embedded software on the MSC system will be presented. It covers the hardware related characteristics like the BSP(Board Support Package), device driver and code patch mechanism.
Keywords: MSC, SBC, BSP, Code Patch, Mission

1. Introduction

The MSC which is a main payload on KOMPSAT-2 satellite is being developed. MSC system consists of EOS(Electro-Optic Subsystem), PMU(Payload Management Subsystem) and PDTS (Payload data Transmission Subsystem), and PMU includes SBC. SBC incorporates Intel 80486 as a main processor. Embedded software on the SBC board communicates with OBC(On-Board Computer) which is a main controller of the spacecraft, and manages all the sub-units of the MSC system. SBC software handles all the information and activities for the mission operation using VxWorks real-time operating system.

2. SBC Software Requirements

SBC software has several functions and capabilities that enable MSC system to perform imaging missions. SBC software receives and executes all the command messages from OBC through mil-std-1553B communication channel. These command messages contain a lot of information for MSC system to execute required missions. SBC software sends SOH(State Of Health) data to OBC every second. This includes not only simple state of all the units but also the important analog telemetry like main power voltage and temperature. SBC software deals with mission execution. MSC system has several predefined missions like real-time imaging, simultaneous imaging and playback, memory playback, OBC data playback, and so on. Every unit should be configured properly according to predefined configuration order before executing a mission. Some missions that shall be executed within at least coming 24 hours will be stored in the SBC memory. SBC software manages several system data tables that are required by different units for their assigned functions. Most of them are to be stored in flash memory and these can be updated to RAM by ground station commands.

3. SBC software design

SBC software incorporates VxWorks as a real-time operating system in order to handle all the simultaneous activities. SBC software consists of four main tasks and several modules to deal with controlling commands and data for imaging and all the state of health telemetry data and to perform interface with the other MSC units. The main structure of SBC software is depicted in fig.1.

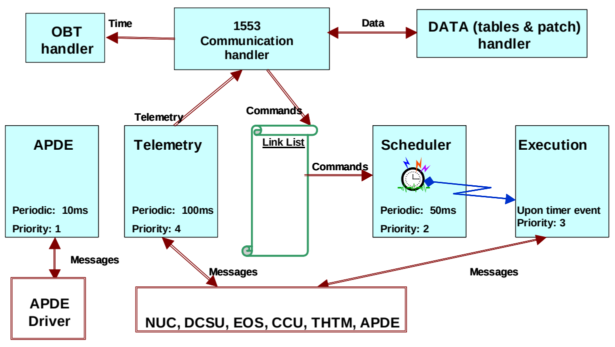

Fig. 1. SBC Software Main Tasks and Main Modules

The 1553 communication handler receives command messages and parses them and transmits them to the appropriate task or module. Spacecraft OBC works as a BC(Bus Controller) and SBC works as an RT(Remote Terminal). The linked list module manages a command table that is sorted by execution time of each command by means of linked list data structure. Each command has its own execution time in half-second. A new command can be inserted into the linked list by the order of execution time and a command in the linked list can be deleted by request. Commands that should be executed in the next delta time can be fetched with respect to the current time. Access to this linked list command table is controlled by mutex semaphore. The scheduler task is required for checking if there are commands that are to be executed and fetching such command from linked list command table. This task will be executed every 100 millisecond. Scheduler task sends each command to the execution task through message queue. The execution task waits for new command from message queue all the time. New command is executed as soon as it arrives. Most of the commands will be executed by sending them to the appropriate units through serial communication channels. Therefore, this task calls suitable communication module to send them. OBC sends time-mark signal every second through discrete line and sends OBT(On-

Board Time) message through mil-std-1553 communication channel every second. There are many system data tables required for mission executions. Data handler module gets parts of or whole table data that should be updated from mil-std-1553B handler module. SBC software supports mission execution. One mission script takes care of performing one whole mission and consists of a series of related commands, which are to be executed in the designated sequence, with a specified time interval between commands. Dozens of mission scripts are defined in advanced and stored to flash memory. Mission script can be activated only when they are copied to the linked list command table by the mission execution command from OBC. Mission execution command will include the index of mission script and the absolute time when the mission should be executed. Designated mission script will be copied to linked list command table and time interval between commands will be translated into the absolute time. The telemetry task gathers all the state of health data from all the units by receiving relevant message through serial communication channel. This task sends telemetry data to OBC every second according to the telemetry format table. The ATS(Antenna Tracking Software) task controls the x-band antenna to communicate with ground station. This task calculates the angle of antenna and sends it to APDE through serial communication channel every 10 milliseconds. In order to calculate the next position of the antenna, the information about the position of satellite in the orbit, the attitude of satellite and the feedback value of current antenna position are required. There are six different software modules to communicate with six different units(EOS, THTM, NUC, APDE, DCSU, CCU). Each communication module deals with transmitting commands and gathering telemetry.

4. SBC Software Initialization

When SBC is boot up, the software follows special sequence because it is operating on the satellite in the space. After turn-on the SBC, the software checks a 'PMU active' discrete signal if current SBC is indicated as an active one. If not, the software will be in idle mode at which nothing will be done. After then, it checks if there is a code patch request. According to the reset type, it performs power-up BIT(Built In Test) which include data bus test, address bus test, RAM test, flash test and 1553 device memory test. It will be done only when it is cold reset, because it is a destructive test. If it is cold reset, the software code in the RAM will be checked using checksum whether there has been SEU(Single Event Upset) and the code is spoiled. If there is code patch request, the part of the operating software in the RAM will be updated from a predefined buffer which has been filled with new software code before previous reset. The general initialization sequence is shown in the fig.2 and code patch mechanism will be described in the next chapter.

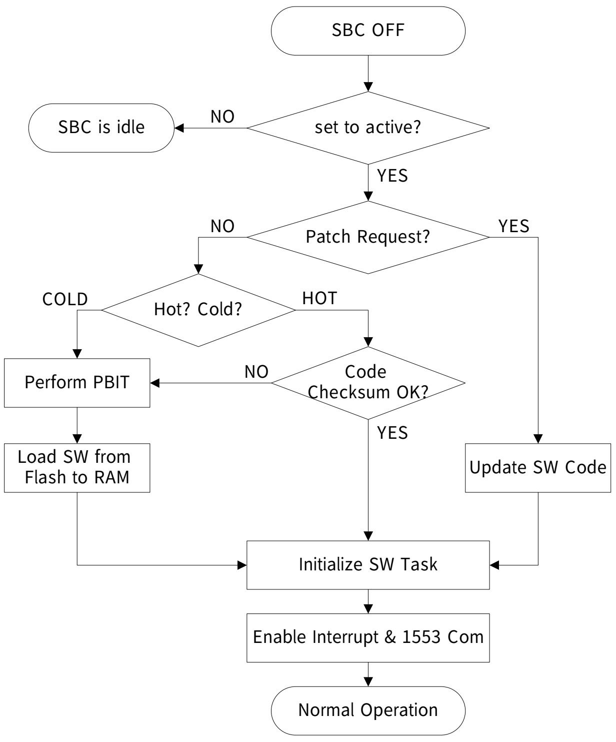

Fig. 2. SBC Software Initialization Sequence

5. BSP & Device Driver

BSP(Board Support Package) has been developed for VxWorks kernel and application software. It was modified from general 80486 BSP to be used for SBC. And it includes x-modem driver in order to establish the communication with host computer. It is managed by full-duplex serial communication and it is managed by interrupt. Device drivers have been developed for 1553 device, UART, discrete signal and system timer and so on. Internal memory and registers of the 1553 devices are mapped as a part of system memory. Because 1553 device has 16 bits data bus, in order to interface 32 bits 80486 CPU, only 4-byte aligned address areas are being used. Device driver handles address mapping from CPU to 1553 device. UART and all the other discrete signals are mapped into IO space. CPU internal cache memory is not being used because it is not immune to the radiation and it is disabled in the BSP. Segmentation and paging schemes are disabled also. Real mode operation of the CPU will be changed to protected mode just after power-on. System clock is adjusted to 100 ticks per second. Boot code, kernel and application will be compressed and reside in the flash memory. When it boots up, they are decompressed and copied into the ram.

6. Code Patch Mechanism

SBC software has been design to be stable, reliable and deterministic and it is being tested very carefully because it will be a part of the satellite. However, after launching KOMPSAT-2 satellite, if some serious bugs are found in the SBC software, they can be fixed using code patch mechanism which was designed and considered during the development phase. It is very unique characteristic of the SBC software compared to the other embedded software. Even though VxWorks provides dynamic linking of the object modules, it is not used for code patch mechanism of the SBC software. All of software code patch mechanism is handled in the application level. SBC software consists of a lot of modules and functions, therefore, various kinds of bugs can be imagined. In order to cover many kinds of possible bugs, code patch mechanism has been designed and implemented very carefully. When some bugs are found, we need to change the software code in order to fix them. At that time, some new local variables may be added, or some new global variables may be added, or some lines of code may need to be added, or some lines of code may need to be deleted, or simply a value of constant may be changed, or some external function may be called. Because the complete software cannot be updated using code patch mechanism, the bug should be fixed with minimum change of software code. When the final SBC software is compiled, linked and programmed into the flash memory of SBC board, the final loadable module and memory map file need to be saved at the ground station because it will be used as a reference for the future code patch. Basic structure of the loadable module of SBC software is described in the fig.3.

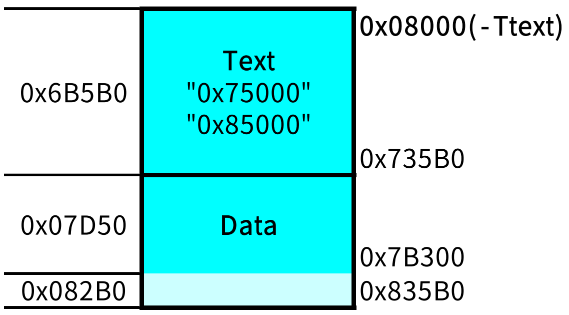

Fig. 3. Structure of Loadable Module(VxWorks)

Output file of the compiler and linker divided into three parts. The ‘text’ section includes all the functions of the software, and the ‘data’ section includes all the global variables which are initialized with some value apparently, and the ‘bss’ section includes un-initialized global variables. That’s why the bss section does not appear in the loadable module which includes relocation information. The loadable module that is described in the fig.3 will be located in the actual SBC RAM after boot-up as described in the fig.4. The memory location of each part can be controlled using link options, -Ttext, -Tdata and –Tbss when we build the loadable module.

In order to fix the bug in the SBC software, some modifications need to be made. Therefore, some part of text, data and bss area may be changed. We can get the exact address of the modification in the SBC RAM by comparing the reference(original) loadable module and the loadable module which was generated with modified SBC software. The code in this memory area can be changed by the ground-station command to MSC system.

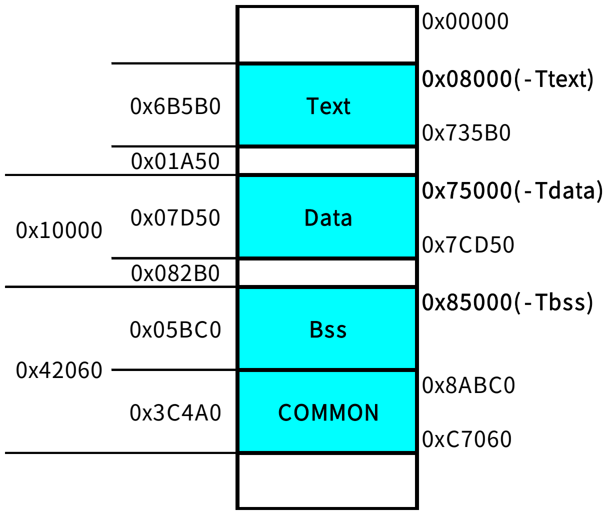

Fig. 4. Software Size and Location in the RAM

In order to minimize the changes of the loadable module, we can insert constant size of gap between every function or between every module. However, if we insert a constant gap between every function, the RAM should be very big. Therefore, only constant size of inter-module gap is inserted in the SBC software. In this case, the change of a certain module will not affect the other module, accordingly, the address of variables and functions in another module will not be changed. There are several kinds of code modifications which cause the changes of pointers of functions and variables in different way. They are changing constant values, adding global variables, adding local variable, deleting one line in the code, adding one line in the code and calling external functions.

7. Conclusion

SBC software has been developed for MSC system to perform ground imaging. It consists of several real-time tasks which make it possible to configure all the sub-units of MSC system in order to provide various mission executions. Even though it has been designed to have high reliability and it will be tested thoroughly, SBC software has a capability of code patch during the on-orbit operation.

References

[1] MSC Critical Design Review Package
[2] SBC Software Requirement Specification
[3] MSC Operation Requirement Specification
[4] MSC PMU Interface Requirement Specification