**Startup Module Description Document**

**1. Introduction**  
The main function of Startup is to define the interrupt vector table, initialize the global variable area, initialize the stack pointer, and then jump to the main function.

**2. Interrupt Vector Table**  
The AP80 series chips have 11 exception interrupts and 16 peripheral interrupts. The first address of the interrupt vector table stores the stack pointer. For details on the interrupt response mechanism, please refer to The Definitive Guide to the ARM Cortex-M3.

The structure of the interrupt vector table is shown in Table 1.

Table 1 Vector Table Structure

|  |  |  |
| --- | --- | --- |
| Number | Table item address offset | Exception type |
| 0 | 0x00 | Initial value of MSP |
| 1 | 0x04 | Reset |
| 2 | 0x08 | NMI |
| 3 | 0x0C | Hard fault |
| 4 | 0x10 | MemManage fault |
| 5 | 0x14 | Bus fault |
| 6 | 0x18 | Usage fault |
| 7-10 | 0x1C-0x28 | Reserved |
| 11 | 0x2C | SVC |
| 12 | 0x30 | Debug monitor |
| 13 | 0x34 | Reserved |
| 14 | 0x38 | PendSV |
| 15 | 0x3C | SysTick |
| 16 | 0x40 | GPIO |
| 17 | 0x44 | RTC |
| 18 | 0x48 | IR |
| 19 | 0x4C | FUART |
| 20 | 0x50 | BUART |
| 21 | 0x54 | PWC |
| 22 | 0x58 | TIMER0 |
| 23 | 0x5C | USB |
| 24 | 0x60 | DMA CH0 |
| 25 | 0x64 | DMA CH1 |
| 26 | 0x68 | DECODER |
| 27 | 0x6C | SPIS |
| 28 | 0x70 | SDIO |
| 29 | 0x74 | SPIM |
| 30 | 0x78 | TIMER0 |
| 31 | 0x7C | WatchDog |

**3. Initializing Global Variables**  
In startup, the initial values of initialized global variables are fetched from Flash, decompressed, and copied into the global variable area. The implementation code is as follows:

229     //get the load region layout table

230     LDR     R4,=|Region$$Table$$Base|

231     LDR     R5,=|Region$$Table$$Limit|

232 \_\_NEXT\_REGION

233     CMP     R4,R5

234     //everything is ok

235     BCS     \_\_REGION\_DECOMP\_OK

236     LDM     R4,{R0-R3}

237     //decompress the data following the compress algorithm as compiling method

238     STMDB   R13!,{R4,R5}

239     ORR     R3,R3,#0x01

240     BLX     R3

241     LDMIA   R13!,{R4,R5}

242     ADD     R4,R4,#0x10

243     B       \_\_NEXT\_REGION

244 \_\_REGION\_DECOMP\_OK

**4. Initializing the Stack Pointer**  
After initializing the global variables, startup initializes the stack pointer and jumps to the main function. The implementation code is as follows:

247     LDR     SP,=\_\_initial\_sp

248     LDR     R0,=main

249     BX      R0