in include/configs/pegmatite.h

```
#define CONFIG_SYS_INIT_SP_SIZE     (2 * 1024)

#ifndef CONFIG_SYS_TEXT_BASE
#define CONFIG_SYS_TEXT_BASE 0x08000000 /* (CONFIG_SYS_BASE_ADDR + CONFIG_SYS_INIT_SP_SIZE + CONFIG_SYS_MALLOC_LEN) */
#endif

#define CONFIG_SYS_BASE_ADDR (CONFIG_SYS_TEXT_BASE - 0x800)
#define CONFIG_SYS_INIT_SP_ADDR (CONFIG_SYS_BASE_ADDR + CONFIG_SYS_INIT_SP_SIZE - GENERATED_GBL_DATA_SIZE)
```

CONFIG_SYS_TEXT_BASE ==> miniloader把u-boot载入到128M的边界。

CONFIG_SYS_BASE_ADDR = 0x08000000 - 0x800

CONFIG_SYS_INIT_SP_ADDR ==> (0x08000000 - 0x800) + 2K - GENERATED_GBL_DATA_SIZE = 0x08000000 - GENERATED_GBL_DATA_SIZE

GENERATED_GBL_DATA_SIZE的定义

in lib/asm-offsets.c

```
DEFINE(GENERATED_GBL_DATA_SIZE,
        (sizeof(struct global_data) + 15) & ~15);
```

---

in arch/arm/lib/crt0.S

```
/*
 * Set up initial C runtime environment and call board_init_f(0).
 */

#if defined(CONFIG_SPL_BUILD) && defined(CONFIG_SPL_STACK)
        ldr     sp, =(CONFIG_SPL_STACK)
#else
        ldr     sp, =(CONFIG_SYS_INIT_SP_ADDR)
                                                                    ①
#endif
        bic     sp, sp, #7      /* 8-byte alignment for ABI compliance */
        sub     sp, sp, #GD_SIZE        /* allocate one GD above SP */
        bic     sp, sp, #7      /* 8-byte alignment for ABI compliance */
        mov     r9, sp          /* GD is above SP */
                                                                        ②
        mov     r0, #0


                                                    ③
        bl      board_init_f
```

①

设置u-boot运行的stack, stack在u-boot code的下面，0x08000000 - GENERATED_GBL_DATA_SIZE

②

把stack top赋值给r9 register。

如果结合arch/arm/include/asm/global_data.h中的如下定义及board_init_f()的code就一目了然了。

in arch/arm/include/asm/global_data.h

```
#ifdef CONFIG_ARM64
#define DECLARE_GLOBAL_DATA_PTR         register volatile gd_t *gd asm ("x18")
#else
#define DECLARE_GLOBAL_DATA_PTR         register volatile gd_t *gd asm ("r9")
#endif
```

in arch/arm/lib/board.c

```
1.  void board_init_f(ulong bootflag)
2.  {
3.          bd_t *bd;
4.          init_fnc_t **init_fnc_ptr;
5.          gd_t *id;
6.          ulong addr, addr_sp;
7.  #ifdef CONFIG_PRAM
8.          ulong reg;
9.  #endif
10.         void *new_fdt = NULL;
11.         size_t fdt_size = 0;
12.
13.         memset((void *)gd, 0, sizeof(gd_t));
                                                              (A)
14.
15.         gd->mon_len = _bss_end_ofs;
                                                                  (
    B)
16. #ifdef CONFIG_OF_EMBED
17.         /* Get a pointer to the FDT */
18.         gd->fdt_blob = _binary_dt_dtb_start;
19. #elif defined CONFIG_OF_SEPARATE
20.         /* FDT is at end of image */
21.         gd->fdt_blob = (void *)(_end_ofs + _TEXT_BASE);
22. #endif
23.         /* Allow the early environment to override the fdt address */
24.         gd->fdt_blob = (void *)getenv_ulong("fdtcontroladdr", 16,
25.                                             (uintptr_t)gd->fdt_blob);
26.
27.  ......
28.
29.  }
```

(A)

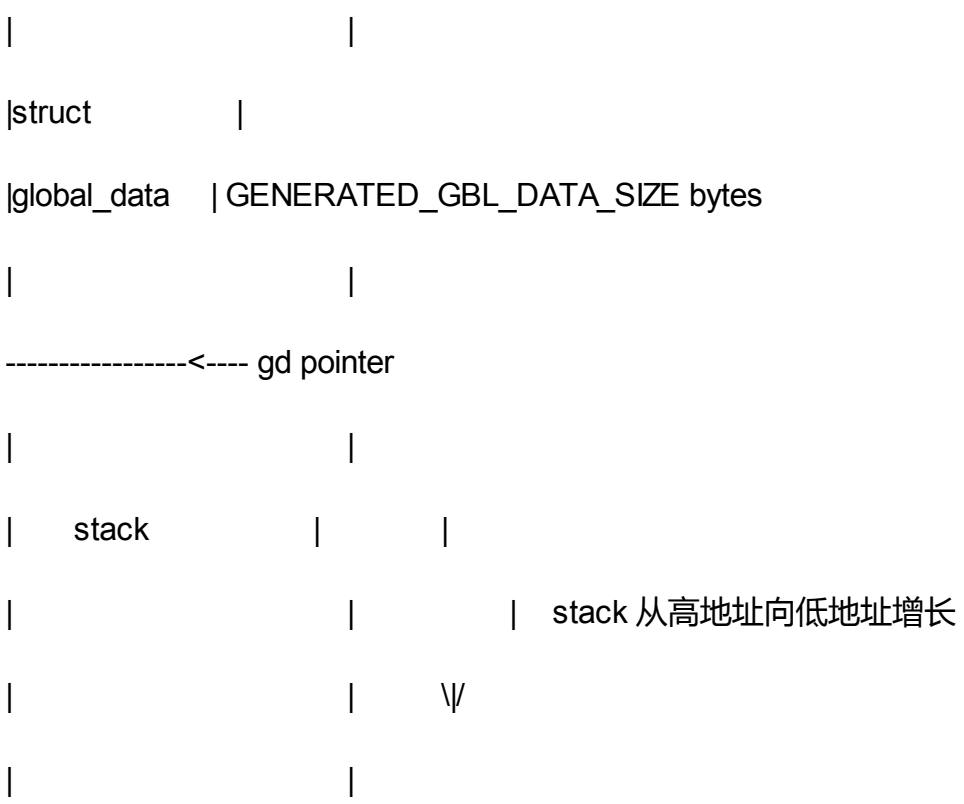
gd variable并没有定义在board_init_f()中。实际上gd就是位于u-boot code和runtime stack之间的那GENERATED_GBL_DATA_SIZE bytes空间,就是struct global_data。

| |

| |

| u-boot code|

----------<---- 128M (0x08000000)



in include/asm-generaic/global_data.h

struct global_data的空间包括如下structure的data。

```
/*
 * The following data structure is placed in some memory which is
 * available very early after boot (like DPRAM on MPC8xx/MPC82xx, or
 * some locked parts of the data cache) to allow for a minimum set of
 * global variables during system initialization (until we have set
 * up the memory controller so that we can use RAM).
 *
 * Keep it *SMALL* and remember to set GENERATED_GBL_DATA_SIZE > sizeof(gd_t)
 *
 * Each architecture has its own private fields. For now all are private
 */

#ifndef __ASSEMBLY__
typedef struct global_data {
        bd_t *bd;
        unsigned long flags;
        unsigned int baudrate;
        unsigned long cpu_clk;  /* CPU clock in Hz!             */
        unsigned long bus_clk;
        /* We cannot bracket this with CONFIG_PCI due to mpc5xxx */
        unsigned long pci_clk;
        unsigned long mem_clk;
#if defined(CONFIG_LCD) || defined(CONFIG_VIDEO) || 1
        unsigned long fb_base;  /* Base address of framebuffer mem */
#endif
#if defined(CONFIG_POST) || defined(CONFIG_LOGBUFFER)
        unsigned long post_log_word;  /* Record POST activities */
        unsigned long post_log_res; /* success of POST test */
        unsigned long post_init_f_time;  /* When post_init_f started */
#endif
#ifdef CONFIG_BOARD_TYPES
        unsigned long board_type;
#endif
        unsigned long have_console;     /* serial_init() was called */
#ifdef CONFIG_PRE_CONSOLE_BUFFER
        unsigned long precon_buf_idx;   /* Pre-Console buffer index */
#endif
#ifdef CONFIG_MODEM_SUPPORT
        unsigned long do_mdm_init;
        unsigned long be_quiet;
#endif
        unsigned long env_addr; /* Address  of Environment struct */
        unsigned long env_valid;        /* Checksum of Environment valid? */

        unsigned long ram_top;  /* Top address of RAM used by U-Boot */

        unsigned long relocaddr;        /* Start address of U-Boot in RAM */
        phys_size_t ram_size;   /* RAM size */
        unsigned long mon_len;  /* monitor len */
        unsigned long irq_sp;           /* irq stack pointer */
        unsigned long start_addr_sp;    /* start_addr_stackpointer */
        unsigned long reloc_off;
        struct global_data *new_gd;     /* relocated global data */
```

```
        const void *fdt_blob;   /* Our device tree, NULL if none */
        void *new_fdt;          /* Relocated FDT */
        unsigned long fdt_size; /* Space reserved for relocated FDT */
        void **jt;              /* jump table */
        char env_buf[32];       /* buffer for getenv() before reloc. */
#ifdef CONFIG_TRACE
        void            *trace_buff;    /* The trace buffer */
#endif
#if defined(CONFIG_SYS_I2C)
        int             cur_i2c_bus;    /* current used i2c bus */
#endif
        unsigned long timebase_h;
        unsigned long timebase_l;
        struct arch_global_data arch;   /* architecture-specific data */
} gd_t;
#endif
```

DEFINE(GENERATED_GBL_DATA_SIZE,

(sizeof(struct global_data) + 15) & ~15);

in include/linux/kbuild.h

#define DEFINE(sym, val) \

asm volatile("\n->" #sym " %0 " #val : : "i" (val))

(B)以后的代码就是初始化struct global_data data了。

③

void board_init_f(ulong bootflag);

mov r0, #0 ==> bootflag = 0