Komunikační rozhraní počítačů

USB periferie

Periferii USB (v našem případě periferii OTG_FS) musíme nejprve inicializovat. Pro tuto akci je třeba mít definované konfigurační registry procesoru. Názvy a adresy registrů a jejich adresy v paměti naleznete v referenční příručce mikrokontroléru. Adresy u jednotlivých registrů jsou uváděny relativně k bázovým adresám USB periferie. Definice proměnné je možná

```
volatile __no_init unsigned long NAZEV@ ADRESA
```

nejedná se ovšem o standardní zápis C, ale zápis pro překladač IAR. Z tohoto důvodu lze použít odlišný přístup prostřednictvím ukazatele na strukturu, který umístíme přímo na bázovou adresu periferie

```
struct struct_usb {
        volatile unsigned long reg1;
        volatile unsigned long reg2;
};
struct struct_usb *usb = USB_BASE_ADDRESS;
usb->reg1 = 0x00;
```

případně přístup definice každého registru pomocí makra

```
#define USB_REG1 (USB_BASE_ADDRESS + 0x0000)
#define USB_REG2 (USB_BASE_ADDRESS + 0x0004)
*( (volatile unsigned long *) USB_REG1) = 0x00;
```

Není třeba definovat všechny registry, ale pouze ty, které budete potřebovat. Periferii USB nakonfigurujeme přiřazením a inicializací vstupně výstupních pinů

```
GPIO_InitStructure.GPIO_Pin =  ??? | ??? | ???;
GPIO_InitStructure.GPIO_Speed = GPIO_Speed_100MHz;
GPIO_InitStructure.GPIO_Mode = GPIO_Mode_AF;
GPIO_InitStructure.GPIO_OType = GPIO_OType_PP;
GPIO_InitStructure.GPIO_PuPd = GPIO_PuPd_NOPULL;
GPIO_Init(GPIOA, &GPIO_InitStructure);

GPIO_PinAFConfig(GPIOA, ???, GPIO_AF_OTG_FS);
GPIO_PinAFConfig(GPIOA, ???, GPIO_AF_OTG_FS);
GPIO_PinAFConfig(GPIOA, ???, GPIO_AF_OTG_FS);
```

Dohromady je třeba nakofigurovat dva (DATA+, DATA-) nebo tři (ještě snímání napětí) piny. Které to jsou, byste měli zjistit z elektrického schématu vývojového kitu a z referenční příručky mikrokontroléru. Dále inicializovat přerušení

```
NVIC_InitStructure.NVIC_IRQChannel = ???;
NVIC_InitStructure.NVIC_IRQChannelPreemptionPriority = ???;
NVIC_InitStructure.NVIC_IRQChannelSubPriority = ???;
NVIC_InitStructure.NVIC_IRQChannelCmd = ???;
NVIC_Init(&NVIC_InitStructure);
```

Dále je možné postupovat podle referenční příručky, kapitoly 29.17 OTG_FS programming model. Po provedení zmíněné inicializace je třeba postupovat podle Core Initialization a dále Device Inicialization. Zde pouze poznámky jako doplnění nejasností z příručky. Pole TRDT se vypočítá jako čtyřnásobek frekvence AHB vydělený 48 MHz (rychlost taktování USB periferie). Toto číslo je ještě třeba zvýšit o jedna. V poli TOCAL nastavíme nejvyšší číselnou hodnotu. Jako nastavení zdrojů přerušení je možné zvolit

```
OTG_FS_GINTMSK = ???;
OTG_FS_GCCFG = ???;
```

Potom již přijdou první dvě přerušení (RESET, ENUMERATION DONE). Dále bude postup dle Device programming model.

Komunikační rozhraní počítačů

Příjem zprávy od hosta

Obslužnou funkci přerušení je vhodné koncipovat následujícím způsobem.

```
void OTG_FS_IRQHandler(void) {
        if (OTG_FS_GINTSTS & ???) {
                OTG_FS_GINTSTS = ???;
                usb_reset();
        }
        if (OTG_FS_GINTSTS & ???) {
                OTG_FS_GINTSTS = ???;
                usb_enum_done();
        }
        if (OTG_FS_GINTSTS & ???) {
                usb_receive();
        }
        return;
}
```

Uvedená kostra funkce slouží pouze jako ukázka, v programu je třeba obsloužit více zdrojů přerušení. Zde by se hodilo říci, že použitý mikroprocesor disponuje dvěma typy přerušení – pulse a level. Z pohledu aplikace se oba typy liší způsobem jejich obsluhy, v případě pulse interruptů je třeba oznámit jejich obsloužení smazáním určitého flagu, pro level interrupty stav daného flagu závisí na stavu dalších registrů. O jaký typ přerušení se jedná je popsáno vždy u příslušného bitu v registru přerušení.

Při vyvolání přerušení RXFLVL (viz referenční příručka) je doporučeno zakázat znovu-vyvolání tohoto přerušení a povolit jej až na konci obslužné rutiny. Dále je třeba vyčíst stav registru OTG_FS_GRXSTSP, který má charakter zásobníku. Je tedy důležité z registru vyčíst data pouze jednou. To zajistíte tím, že vytvoříte proměnnou, do které obsah tohoto registru zkopírujete pomocí přiřazení „=“ a dále budete pracovat s daty pouze nad vlastní proměnnou. Ze zmíněného registru je možné vyčíst, jaký počet bytů přišel a také o jaký typ operace (STATUS, OUT) se jedná.

Struktura příchozích zpráv (BM REQUEST) je popsána v dokumentu USB Specification (http://www.usb.org) v sekci pro vývojáře. Pro účely cvičení je možné použít specifikaci pro verzi USB 1.1 nebo 2.0. V těchto dokumentech se nachází i popis požadovaných deskriptorů.

Ohledně aplikační vrstvy je možné implementovat třídu zařízení dle volby. Seznam tříd včetně dodatečné dokumentace je na adrese

http://www.usb.org/developers/devclass_docs#approved

Jednodušší aplikace je implementace zařízení dle HID, dále je doporučeno implementovat např. zařízení dle CDC.