Embedded systémy

Obsah

1 Přerušení programu

2 Vektorová tabulka

Vektorová tabulka je kritická datová struktura v paměti mikrokontroléru, která slouží jako most mezi hardwarem a softwarovou obsluhou událostí. Je to pole ukazatelů, z nichž každý ukazuje na specifickou funkci, která se nazývá **obsluha (handler)**. Tyto handlery se spouštějí v reakci na události, jako je reset procesoru, systémová chyba nebo přerušení od periférie.

Koncept vektorové tabulky je platformně závislý, to znamená, že každý kontroler nebo procesor bude obsahovat jiný mechanismus fungování vektorové tabulky, ale základní pricip je většinou stejný.

2.1 Jak to funguje

Při výskytu události, kterou procesor vyžaduje, jako například externí přerušení (například stisk tlačítka), procesor okamžitě přeruší svou aktuální činnost. Místo toho, aby se snažil událost vyřešit sám, podívá se do své vektorové tabulky. Podle typu události najde v tabulce příslušnou adresu handleru a skočí na tuto adresu, aby provedl příslušnou funkci.

Jakmile je obslužná funkce dokončena, procesor se vrátí a pokračuje v původní činnosti, jako by se nic nestalo. Tento mechanismus umožňuje efektivní multitasking, aniž by procesor musel neustále kontrolovat stav každé periferie.

2.2 Obecná struktura

Přesná struktura a pořadí záznamů ve vektorové tabulce se liší v závislosti na konkrétní architektuře (např. ARM, RISC-V, AVR). Většina tabulek však obvykle zahrnuje následující typy záznamů:

- **Vstupní bod programu**: Ukazatel na první instrukci, která se má provést po spuštění nebo resetu systému. V některých architekturách je to přímo adresa reset handleru, v jiných je to adresa zásobníku.
- **Systémové obsluhy**: Adresy funkcí pro obsluhu systémových událostí nebo chyb, jako je například dělení nulou.
- **Obsluhy přerušení**: Adresy pro obsluhu přerušení od různých periferií, jako je například UART, SPI, časovač nebo AD převodník.

Většina procesorů má vektorovou tabulku umístěnou na pevně dané adrese v paměti (obvykle na jejím začátku), aby ji procesor mohl snadno najít ihned po zapnutí.

2.3 Vektorová tabulka v kontextu programování

Úkolem systémového programátora je vyplnit tuto tabulku správnými adresami. To znamená napsat obslužné funkce pro události, na které je reagovat, a pak zajistit, aby linker správně umístil adresu těchto funkcí do příslušných pozic ve vektorové tabulce. Bez správně nakonfigurované a umístěné vektorové tabulky by procesor nebyl schopen spustit program ani reagovat na externí události, což by vedlo k nefunkčnímu systému.

Pro nastavení kde má být v paměti procesoru/kontroleru vektorová tabulka umístěná slouží **linker script**.

3 Linker script

Linker skript je textový soubor, který řídí, jak linker (program, který spojuje zkompilovaný kód a knihovny do spustitelného souboru) uspořádá různé části (sekce) výsledného programu v paměti mikrokontroléru. V embedded programování je to klíčový nástroj, protože vám umožňuje přesně definovat, kam se kód a data uloží, což je nezbytné pro efektivní využití omezené paměti.

3.1 Základní koncepty a struktura

Linker skript se typicky skládá ze tří hlavních částí: MEMORY, SECTIONS a ENTRY.

3.1.1 MEMORY

Tato sekce popisuje fyzické paměťové oblasti, které má mikrokontrolér k dispozici, jako je FLASH (pro programový kód a konstanty) a RAM (pro proměnné a zásobník). Pro každou oblast definujete její název, počáteční adresu (ORIGIN) a velikost (LENGTH).

Například:

```
MEMORY
{
  FLASH (rx) : ORIGIN = 0x08000000, LENGTH = 128K
  RAM (rwx) : ORIGIN = 0x20000000, LENGTH = 20K
}
```

V tomto příkladu má FLASH počáteční adresu 0x08000000 a velikost 128K, a RAM začíná na 0x20000000 s velikostí 20K. Atributy jako *(rx)* a *(rwx)* určují, zda je paměťová oblast čitelná, zapisovatelná nebo spustitelná.

Obecně se příkaz zapisuje takto:

```
MEMORY
  {
    name [(attr)] : ORIGIN = origin, LENGTH = len
    ...
  }
```

Popisek "name" odkazuje na název regionu používaný v linker scriptu. Nastavení "attr" je volitený seznam atributů, které nastavují vlastnosti daného sektoru. Tyto parametry mohou být:

- 'r' - oddíl pouze pro čtení, například pro FLASH kam se zapisuje seznam instrukcí tvořící program.
- 'w' - oddíl pro čtení i zápis, například oddíl RAM, který uchovává data, která vznikají za běhu programu
- 'x' - oddíl může obsahovat instrukce programu, které mohou být vykonávány procesorem
- 'a' - alokovatelná sekce
- 'i' - inicializovaná sekce
- '!' - obrátí platnost každé atributu, který následuje

3.1.2 SECTIONS

Toto je nejdůležitější část. Popisuje, jak linker přebírá vstupní sekce z objektových souborů (zkompilovaný kód) a spojí je do výstupních sekcí ve výsledném spustitelném souboru. Každý objektový soubor vytvořený kompilátorem obsahuje své vlastní sekce (například .text pro kód, .data pro inicializovaná data a .bss pro neinicializovaná data). Funkce

linkeru je sloučit všechny tyto vstupní sekce ze všech objektových souborů do jedné jediné výstupní sekce ve finálním spustitelném souboru. Linker skript mu pak k tomu dává přesné pokyny jak to má udělat.

Standardní sekce zahrnují:

- **.text**: Obsahuje kód programu. Typicky se umisťuje do FLASH paměti, protože se nemusí měnit za běhu, ale data se musejí uchovávat i po vypnutí napájení.
- **.rodata**: Obsahuje data pouze pro čtení (např. konstanty, textové řetězce). Také se umisťuje do FLASH.
- **.data**: Obsahuje inicializované globální proměnné. Tyto proměnné se sice za běhu mění a jsou proto v RAM, ale jejich počáteční hodnoty musí být uloženy ve FLASH paměti, aby se po startu mikrokontroléru mohly zkopírovat.
- **.bss**: Obsahuje neinicializované globální proměnné. Na začátku programu se tato oblast vynuluje. Tyto proměnné existují pouze v RAM a nezabírají místo ve FLASH.

3.1.3 ENTRY

Tento příkaz definuje vstupní bod do programu, tedy první instrukci, která se má provést po spuštění mikrokontroléru.

3.2 Bootloader