Dokumentácia z predmetu IMP: ARM/FITkit3: Zabezpečenie dát pomocou 16/32-bit. kódu CRC

Meno a priezvisko: Juraj Ondrej Dúbrava
Login: xdubra03

Tento text slúži ako dokumentácia k projektu z predmetu IMP a opisuje riešenie projektu zabezpečenia dát pomocou 16/32 bit kódu CRC na platforme ARM/FITkit3.

CRC (Cyclic redundancy check)

CRC je kód slúžiaci na detekciu chýb v prenášaných dátach, napríklad detekciu bitových chýb pri prenose dát po sieti. Pre dáta sa vypočíta hodnota kontrolného súčtu, ktorá sa pripojí k dátam. Hodnota sa vypočíta delením, zvyšok po delení tvorí kontrolnú hodnotu.

Implementácia

Na základe zadania bolo potrebné vytvoriť blok testovacích dát a vypočítať pre nich 16 a 32-bitový CRC kód pomocou HW modulu, pomocou hodnoty z CRC tabuľky a pomocou základného algoritmu.
Ako testovacie dáta boli zvolené dáta náhodným generovaním o dĺžke 512 bajtov. CRC-16 aj CRC-32 sa používajú na zabezpečovanie dlhších dát, preto bola zvolená takáto dĺžka.
Pre výpočet hodnoty pomocou HW modulu bolo využité API z MCUXpresso SDK. Pre 16-bitovú verziu bol použitý CRC-16 s použitým polynómom 0x8005. Ako počiatočná hodnota bola použitá hodnota 0, otáčanie vstupu a výsledku boli vypnuté. Pre 32-bitovú verziu bola použitá verzia CRC-32 s polynómom 0x04C11DB7. Ako počiatočná hodnota nebola zvolená typická hodnota 0xFFFFFFFF, ale hodnota 0 a otáčanie vstupu a výsledku boli vypnuté. Výpočet môže byť uskutočnený aj pomocou iných CRC algoritmov, ktoré sa líšia parametrami, ale na demonštráciu boli vybrané práve tieto 2 spôsoby.

Ďalším spôsobom výpočtu CRC bolo použitie hodnôt z tabuliek. Pre verziu CRC-16 bola vytvorená tabuľka pomocou funkcie *computeCrc16Table* s pomocou použitia polynómu 0x8005 s 256 hodnotami a rovnako tabuľka pre CRC-32 pomocou funkcie *computeCrc32Table*, ale za použitia polynómu 0x04C11DB7. Tieto tabuľky obsahujú hodnoty CRC po delení každej možnej hodnoty bajtu (preto 256 hodnôt tabuľky) daným polynómom. Indexom tabuľky je hodnota deliteľa, ktorým je hodnota bajtu. Výsledkom je zvyšok pre daný bajt. Tabuľky slúžia na predpočítanie týchto hodnôt na urýchlenie výpočtov CRC hodnoty. Iným spôsobom práce s tabuľkou by bolo staticky ju definovať v kóde, čo má však podľa rôznych zdrojov vyššiu réžiu pri načítaní z pamäte a výpočet tabuľky sa uvádza v niektorých prípadoch ako rýchlejší. Po vytvorení tabuliek boli následne zavolané funkcie *computeCrc16* a *computeCrc32* na výpočet CRC hodnôt pre testovacie dáta. Na výpočet tabuliek aj hodnôt CRC boli použité algoritmy z http://www.sunshine2k.de/articles/coding/crc/understanding_crc.html

Posledným spôsobom výpočtu bolo použitie základného algoritmu na výpočet CRC hodnôt. Verzia CRC-16 aj CRC-32 pracujú priamo s bajtami a nie len po jednotlivých bitoch. Na výpočet CRC hodnôt boli opäť použité rovnaké polynómy ako v predchádzajúcich spôsoboch. Výpočet prebieha pomocou funkcií *CRC16_Simple* a *CRC32_Simple*.

Algoritmus môže popísať nasledovne: v každom kroku algoritmu je potrebné zarovnať počiatočnú jednotku polynómu s prvou najvýznamnejšou jednotkou v spracovávaných dátach. Ak je najvyšší bit dát 1, je potrebné uskutočniť bitový posun vľavo a delenie pomocou operácie xor s daným polynómom, inak len uskutočníme bitový posun vľavo o 1 bit.

Po každom výpočte CRC hodnoty pre daný spôsob bola táto hodnota porovnaná s napred definovanou vypočítanou hodnotou CRC pre dané testovacie dáta. Ak sa hodnota zhoduje, vypíše sa hláška o zhode, inak sa vypíše chyba. Výpis je realizovaný na terminál pomocou sérovej komunikácie pomocou modulu *UART* s využitím pollingu.

Možnosti detekcie chýb

Dĺžka CRC hodnoty je dôležitá z hľadiska detekcie chýb. Čím viac bitov CRC, tým menšia je pravdepodobnosť kolízie, čím je myslené to, že rôzne dáta budú mať rovnakú hodnotu CRC. Sila CRC závisí od stupňa použitého polynómu a takisto od samotného polynómu. Pre CRC-16 sa často používajú polynómy 0x8005 a 0x1021, pre CRC-32 polynóm 0x04C11DB7. Hardvérové aj softvérové riešenie musí byť schopné pri použití rovnakého polynómu detekovať rovnaké chyby v dátach. CRC je schopné detekovať všetky jednochyby.
Je schopné detekovať aj všetky dvojchyby, ktoré spĺňajú podmienky, že ich vzdialenosť je menšia ako stupeň použitého polynómu. Rozpoznané budú aj chyby s nepárnym počtom chybných bitov ak sa použije polynóm, ktorý je násobkom x+1. CRC detekuje aj tzv. burst chyby. Všetky burst chyby dĺžky n budú detekované polynómom stupňa n.
Pri použití CRC-16 je pravdepodobnosť, že pri zmenení zabezpečených dát nebude detekovaná chyba $1/2^{16}$ oproti $1/ 2^{32}$ pri CRC-32.

Realizačná a výpočetná réžia

Výpočetná réžia je medzi softvérovým a hardvérovým riešením veľmi dôležitá. Hardvérový CRC modul dokáže byť značne rýchlejší oproti softvérovému výpočtu. Softvérový výpočet vyžaduje niekoľko stoviek cyklov zbernice (približne 700) na spracovanie 1 bajtu v algoritme. Ako príklad hardvérového riešenia je uvedený hardvérový CRC modul Flexis AC CRC na MC9S08AC128.[1] Tento modul umožňuje posun bajtu v 1 cykle zbernice, čo je výrazné zrýchlenie. S týmto modulom bol uskutočnený výpočet CRC16-CCITT na 128 KB dát. Hardvérový modul výpočet zvládol za 170 ms oproti 6.7 s, ktoré potreboval softvér. Takéto zrýchlenie je už výrazne citeľné.
Ak sa nepoužije hardvérový modul na výpočet CRC hodnoty, základný algoritmus môže byť zrýchlený aj iným spôsobom. Týmto spôsobom je vytvorenie CRC tabuľky, ktorá obsahuje hodnoty zvyšku po delení zvoleným polynómom. CRC tabuľka s 256 hodnotami, to znamená pre hodnotu každého možného bajtu, dokáže zrýchliť výpočet až 4-násobne a ak sa zvolí vhodný polynóm tak až 8-násobne[2]. Tabuľka môže mať aj viac ako 256 hodnôt, ak sa bude spracovávať viac ako 1 bajt naraz. Tabuľka takisto môže byť statická alebo sa vytvorí až na začiatku programu. Načítanie tabuľky z pamäti sa v niektorých prípadoch uvádza ako pomalšie riešenie ako jej dynamické vytvorenie. Najpomalším spôsobom výpočtu je teda základný algoritmus.

[1] https://www.nxp.com/docs/en/application-note/AN3795.pdf
[2] https://users.ece.cmu.edu/~koopman/pubs/KoopmanCRCWebinar9May2012.pdf