FAKULTA APLIKOVANÝCH VĚD
ZÁPADOČESKÉ UNIVERZITY
V PLZNI

Simulátor flash paměti pro embedded zařízení

Oborový projekt

Jaroslav Rosa

29. června 2023

Obsah

Zadání

Zadáním práce bylo vytvořit simulátor flash paměti, která umožní komunikaci se samotnou pamětí za pomoci instrukcí, které ji budou ovládat. Paměť bude moct být definována pomocí parametrů, zadaných uživatelem. Simulátor umožní jak zapisovat a číst stránky, mazat bloky, tak měnit některé parametry za běhu, nebo uložit a nahrát obsah a stav paměti.

Analýza

Flash paměť je extrémně rychlá, nevolatilní paměť, která umožňuje do základní jednotky, stránky, zapsat pouze jedenkrát. Před dalším zápisem je potřeba část paměti nejdříve vymazat. Paměť se skládá ze tří základních částí, a to bloku, stránky případně sektoru. Blok je nejmenší mazatelná část paměti, jež se skládá z několika stránek, zapisovatelných částí. Stránka se může následně skládat z několika sektorů. Názvy částí a jejich složení a parametry se však mohou lišit výrobcem. Pro funkčnost simulátoru budou definovány všechny tři. Blok a stránku bude možné definovat pomocí vstupů od uživatele. Dále bude možné definovat počet stránek. Z toho se odvodí počet bloků a sektorů, ty budou mít pevně danou velikost 512B. Jak pro jednotlivé jednotky paměti, tak pro paměť samotnou budou ukládána statistická data, jako čas posldního zápisu, celkový čas zápisu nebo jakýsi reálný čas, tedy součet časů všech operací nad pamětí. S pamětí bude možné provádět hned několik operací, jak základní mazání bloků, zápis a čtení celých stránek a sektorů, tak uložení a načtení dat a stavu paměti.

Sektor

Sektor je nejmenší uskupení bytů paměti. Jeho velikost dat je pro jednoduchost zvolena na 512 bytů, případně je možné ji přepsat změnou konstanty *DEFAULT_SECTOR_SIZE*. Sektor obsahuje krom datové části taktéž část pro ECC, tedy zabezpečení proti chybám v datech. Chyby paměti jsou simulovány za pomocí automatické funkce `Simulate_Error`, která přepisuje obsah ECC a tím generuje chyby. Dále sektor obsahuje bit pro určení, zda-li je v něm již zapsáno a bit pro určení chyby v ECC.

Stránka

Stránka je ucelená jednotka, která obsahuje několik sektorů. K obsahu sektoru přidává data navíc pro její správu jako je bit pro určení poškozené stránky.

Blok

Blok je nejmeněí mazatelní jednotka, která v sobé obsahuje několik stránek a data navíc, která určují stav samotného bloku. Mezi ně patří např. bit pro určení bloku, ktrey již nelze použit. Na jednotlivé stránky paměti se postupně zapisuje, násdledně se pak obsah všech stránek vymaže, a to právě pomocí vymazání celého bloku. Nevýhoda Flash pamětí je ta, že mají omezený počet

mazání jednotlivých bloků, proto je důležité zaznamenávat počet mazání aby bylo následně možné optimalizovat algoritmy pro lepší využití paměti. Samotná paměť nakonec obsahuje několik bloků.

Implementace

Program byl implementován v jazyce C++ a sestaven za pomocí CMake. Je rozdělen do tří částí, a to samotný simulátor, třídu **Flash_Memory**, třídu **Test_Simulation**, která umožňuje uživateli interagovat se simulátorem za pomocí příkazové řádky a soubor pro testování simulátoru **test.hpp**

Simulátor

Flash simulátor je tvořen hlavní třídou **Flash_Memory**, která slouží jako rozhraní pro případné použití simulátoru jako knihovny. Třída implementuje všechny metody pro obsluhu jednotlivých instrukcí paměti. Dále je simulátor tvořen statistickými třídami pro bloky, stránky a celou paměť. Jednotlivé implementace jsou komentovány v kódu.

Cache

Velikost cache paměti je stejná jako datová část stránky. Cache slouží pro vkládání a načítání dat z paměti.

Status registr

Status registr slouží k ukládání stavu proběhlých instrukcí paměti. Jednotlivé bity jsou vysvětleny níže.

Číslo bitu	0	1	2	3	4	5	6	7
Význam	WP	BP	EPE	RB	ES	PT	ECC	ERR

- WP - Write Protect: Indikuje ochranu paměti před zápisem.
- BP - Block Protect: Indikuje ochranu daného bloku před zápisem.
- EPE - Erase Program Error: Indikuje chybu při poslední operaci mazání nebo zápis.
- RB - Ready/Busy: Indikuje, že pamět právě provádí operaci nebo že je připravena k použití.

- ES - Erase Suspend: Indikuje pozastavení možnosti mazání bloku např. pokud má paměť malý přísun energie.
- PT - Programming Time-out: Indikuje timeout u poslední operace zápisu.
- ECC - ECC/Checksum: Indikuje chybu ECC nebo checksumy.
- ERR - Error: Indikuje ostatní chyby.

Adresování

Pro adresování bylo zvoleno jedno číslo ve tvaru 32 bitového integeru. To představuje tři adresy pro jednotlivé jednotky. Adresa bloku představuje prvních 16 bitů čísla, dalších 8 bitů představuje adresu stránky a posledních 8 bitů představuje adresu sektoru. Takto lze zadat adresu například tímto způsobem: 0x00010203, kdy je přistupováno na druhý blok, třetí stránku, čtvrtý sektor.

Automatické úlohy

Pro simulování používání a s tím spjaté chybovosti a změnou času běhu instrukcí paměti bylo zapotřebí implementovat automatické funkce, které budou tyto parametry měnit, resp. generovat chyby. Dle např. této studie bylo zjištěno, že k největší chybovosti dochází při zápisu a čtení z paměti. Proto je funkce pro chybovost funkcí pravděpodobnosti, že dojde k chybě. Dle další studie bylo zjištěno, že čas při čtení se nijak zvlášť nemění, změna času zápisu se snižuje vždy s počtem zápisů dané stránky a čas mazání bloku se zvětšuje s počtem mazání daného bloku.

Třída integrující simulátor

Pro prezentaci funkčnosti simulátoru paměti byla vytvořena obalovací třída **Test_Simulation**, která slouží k interakci uživatele se simulátorem. Nejdříve vezme parametry zadané uživatelem a podle nich nakonfiguruje simulátor. Následně pomocí příkazového řádku naslouchá, spouští a nakonec vypíše výstup instrukcí paměti. Vstup a výstup lze taktéž zaměnit za soubory.

Testy

V čásrti pro testy jsou postupně testovány všechny instrukce paměti. Test je spuštěn před spuštěním samotného simulátoru. Jeho výstup je tedy vidět na začátku běhu.

Uživatelská příručka

Kód lze zkompilovat za pomocí programu CMake. Dále lze spustit za pomocí běžného programu.

Při spuštění lze přidat pomocí znaku '-' parametrizovat jednotlivé hodnoty:

- block_size - velikost bloku
- page_size - velikost stránky
- num_of_block - počet bloků
- mem_type - typ paměti - slc, mlc, tlc, qlc
- read_page_time - čas čtení
- page_prog_time - čas programování
- erase_time - čas mazání
- com_time - simulace času komunikace s pamětí
- max_erase_unm - maximální počet mazání bloku
- input_file - vstupní soubor
- output_file - výstupní soubor

Závěr

Výsledný simulátor je schopný přijímat a zpracovávat příkazy podobně jako reálná paměť. Taktéž je schopná uchovávat statistická data o paměti. Simulátor je možné rozšířit například o věrohodnější automatické funkce nebo další cache, která by umožňovala čtení celého obsahu jednotky a tak zpracovávat všechna její data.