Vysoké učení technické v Brně
Fakulta informačních technologií

IMP-Mikroprocesorové a vestavěné systémy

2018/2019

ARM-FITkit3: Aplikace modulu RNGA

Hynek Bernard
xberna16
17.12.2018

vedoucí projektu
J. Strnadel

Obsah

1 Úvod

1.1 Zadání

Seznamte se s principem tvorby vestavných aplikací v jazyce C založených na mikrokontroléru Kinetis K60 (s jádrem ARM Cortex-M4) fy Freescale v prostředí Kinetis Design Studio (KDS) nebo MCUXpresso.
V jazyce C (tak i dále) vytvořte projekt demonstrující možnosti modulu Random Number Generator Accelerator (RNGA) dostupného na čipu Kinetis K60 z desky platformy FITkit 3.
UPŘESNĚNÍ: 1) Vytvořte aplikaci umožňující generování náhodných čísel pomocí RNGA.

2) Uložte dostatečně reprezentativní množinu posloupností čísel generovaných modulem RNGA.

3) Na základě získaných posloupností vhodně analyzujte a zdokumentujte vybrané vlastnosti generovaných čísel, např. determinismus, délku cyklu, jednotnost, korelaci.

4) Identifikujte slabiny generátoru, navrhněte a implementujte mechanismus s cílem zmírnit zvolenou slabinu.

5) Schopnost mechanismu zmírnit zvolenou slabinu experimentálně prokažte.

1.2 FitKit3

FitKit3 je samostatný hardware, který obsahuje výkonný mikrokontrolér určený pro výuku studentů VUT FIT. Byl mi zapůjčen fakultou. Software se na mikrokontroler tvoří v jazyce C

1.3 RNGA

RNGA modul je digitální integrovaný obvod generující 32-bitová náhodná čísla. Je umístěný v mikrokontroleru K60DN512M10, který je součástí FitKitu3. Konfigurace registrů zajišťuje statisticky dobrá data, to znamená, že data vypadjí náhodně. Neexistuje žádný kryptografický důkaz o bezpečnosti takové generace náhodných dat[1]. RNGA disponuje Entropy registrem, to znamená, že pro náhodná data se může vygenerovat tzv entropy (náhodná veličina) která určí výsledek dalších dat. Nejlepší je tuto hodnotu brát najednou z co nejvíce proměnlivých a těžko reverzně dohledatelných zdrojů (pohyb myši, přesný čas atd..)

2 Návod

2.1 Požadavky

Mnou naprogramovaný fitkit komunikuje přes sériové rozhraní (COM port). Proto je zapotřebí programu který dokáže data z linky číst (například PuTTy). V případě potřeby programování fitkitu je nutné použít Kinetis Design Studio. Jiné utility nejsou potřeba

2.2 Nastavení

Parametry pro sériovou linku jsou : Rychlost- 115200, data bits- 8, stop bits-1, parity- none, flow control- none

2.3 Spuštění

Po připojení stačí stisknout tlačítko SW3 na fitkitu (DOWN) a začnou se ihned generovat data, která se posílají na sériovou linku oddělena mezerou. Rovněž se rozsvítí dioda D12, která značí že program běží a dioda D9, která značí rozsvícením a zhasnutím ukončení výpisu čísla, začne problikávat. Běh se dá kdykoliv pozastavit znovustisknutím tlačítka SW3. Pro přidání algoritmu plnění entropy registru RNGA je možné stistknout tlačítko SW5 (UP), přičemž se rozsvítí dioda D10. Doplňující algoritmus se dá kdykoliv vypnout znovustisknutím tlačítka SW5.

3 Implementace

3.1 RNGA

Pro zprovoznění rgna je pro něj zapotřebí nastavit hodiny, to dělám ve funkci PortsInit(). Poté již jen nastavím ve funkci RNGInit() kontrolnímu registru hodnotu 0xB, čímž vyčistím a maskuji přerušení, LSB značí spuštění RNGA modulu. Poté se počká na inicializaci oscilátorů (funkce delay) a pokud není program pozastaven, začne generovat čísla a vypisovat je. Pokud je entropy=1 tak je ve funkci printInt() plněn registr entropie následujícím způsobem:

Představme si že indexujeme náhodně vygenerované číslo zprava (pro číslo 3587250926 je číslo 6 na nultém indexu a 3 na devátém indexu)

První vyplnění entropy registru posune právě vygenerované číslo o součet hodnot nultého a prvního indexu, v tomto případě o 8

Druhé vyplnění posune číslo o hodnotu prvního a nultého indexu, tedy o 26

Třetí vyplnění posune číslo o hodnotu druhého indexu, tedy o 9

Čtvrté vyplnění posune číslo o hodnotu třetího indexu, tedy o 0

Výše uvedený algoritmus je moje vylepšení stávajícího, výsledky se neliší výrazně, ale množina vygenerovaná s přidáním mého algoritmu prošla testy, kterými neprošla množina bez algoritmu. Dále by se dal algoritmus upravit například přidáním konkrétního timestampu s co nejvyšší přesností.

Rychlost generování dat do output registru nemusíme řešit, protože doba za kterou se vypíše číslo (převod na chary a odeslání UARTEM) je mnohonásobně větší než doba pro vygenerování nové pseudonáhodné hodnoty.

3.2 UART5

Pro zprovoznění UART5 je také zapotřebí nastavit hodiny + inicializovat port pro vysílač (PORTE[8]), taktéž je nastavení provedeno ve funkci PortsInit(). Další nastavení probíhá ve funkci UARTInit(). Nejprve je nutné vypnout transmitter i receiver v registru C2. Nastavuji v registru BDL rychlost 115 200Bd a 1stop bit hodnotou 0x1A. Nuluji adresy registrů C1,C3,MA1,MA2,BDH, jelikož nevyužíváme matchování adres a používáme 8 data bitů bez parity. Nakonec v registru C2 nastavíme hodnotu 0x8 která zapíná vysílač, jaký je od této chvíle funkční.

3.3 Diody a tlačítka

Diody a tlačítka jsou nastavovány ve funkci PortsInit(), kde diody jsou na portuB a tlačítka na portuE. Pro svůj kód jsem se inspiroval cvičeními z laboratoří a tlačítka jsou konfigurována identicky jako v laboratoři číslo 2 (s přenesením na fitkit3, což zahrnuje změnu názvu portu podle schéma) pull up rezistory. Rovněž se pro tlačítka musí zapnout obsluha přerušení, aby se jimi dal program ovládat za běhu.
Pro tlačítka bylo nutné vytvořit handler přerušení, ve kterém se mění stavy diod D10 a D12 podle hodnoty kterou tlačítka nastavují (SW5 a D10 úprava algoritmu, SW3 a D12 spouštění generátoru). Dioda D9 označuje vypsání čísla na výstup a není s žádným tlačítkem v handleru spjata. Funkci diody D9 je možné vidět při průběhu přerušení kdy dioda na chvíli přestává svítit protože neprobíhá žádný výpis.

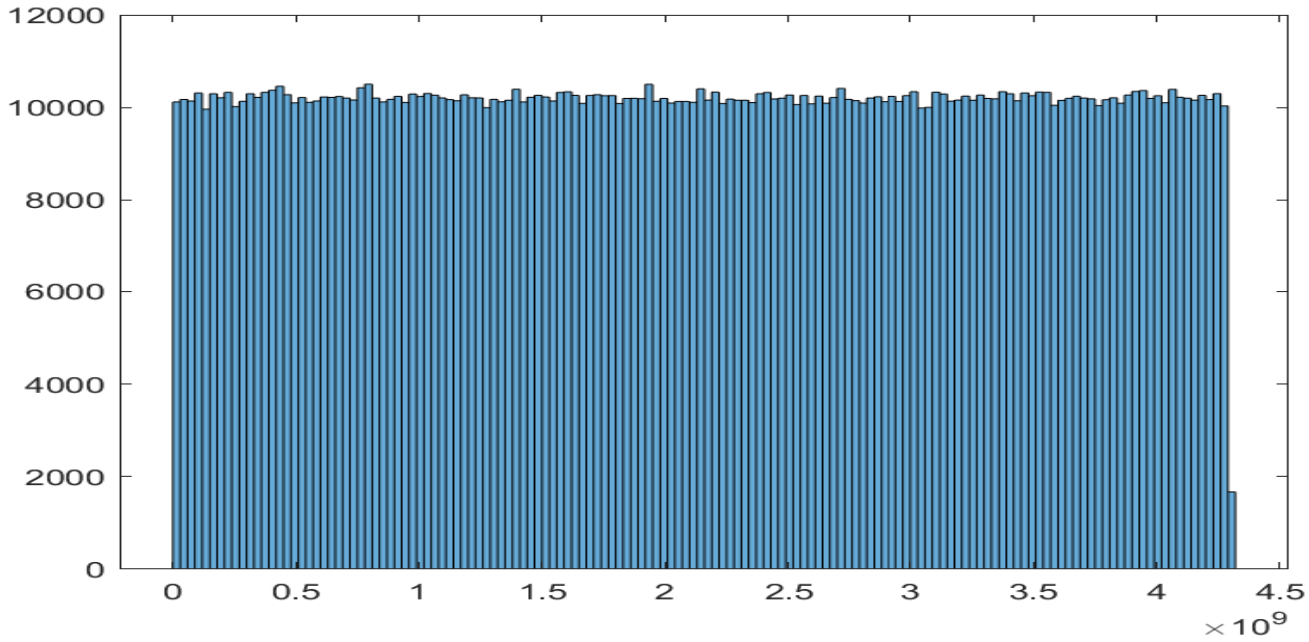
3.4 Program

Program je zapsán tak, aby stisk tlačítek nenarušoval výpis. Tlačítka upravují hodnoty boolů které určují v podmínkách co se má dále dít. Pro korektní výpis je vygenerované číslo z RNGA ve funkci printInt() bráno jak integer bez znaménka (32 bitové nezáporné číslo) které se rozebere po jednotlivých cifrách na chary a vyšle se na UART. Program je v nekonečném cyklu, aby se zamezilo ukončení.

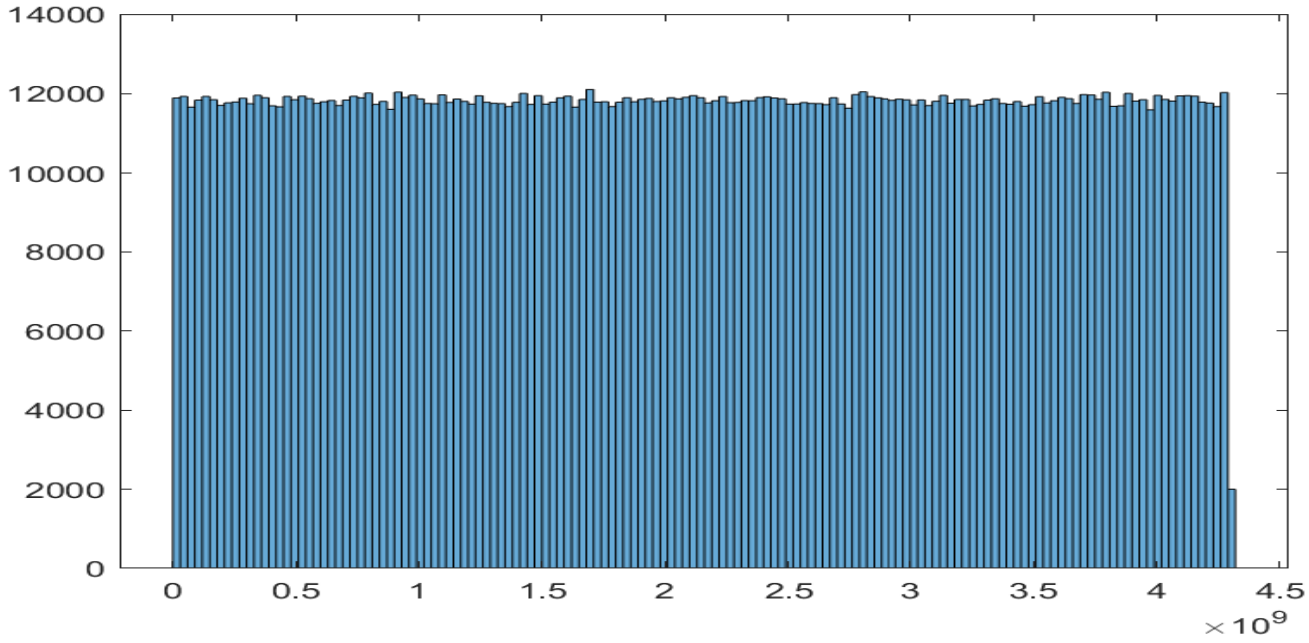
4 Závěr

4.1 Analýza generovaných dat

Vygeneroval jsem nejprve množinu dat A s počtem 1461651 vzorků, poté množinu dat B s počtem 1694342 vzorků. Množina A byla generována bez entropy veličiny a množina B měla po celou dobu zapnutý můj entropy algoritmus.
V matlabu vizuálním porovnáním histogramů veličin vidíme, že hodnoty v množině B jsou rovnomněrněji rozložené narozdíl od A. U obou množin je poslední sloupec malý kvuli omezení velikosti uint32.



Obrázek 1: Histogram množiny A



Obrázek 2: Histogram množiny B

Další analýza proběhla opensourcovým programem pro analýzu generátorů náhodných čísel dieharder. Použil jsem verzi 3.31.1 subsystémem Ubuntu ve Windows 10. Vstupní soubory veličin potřebovaly úpravu (převedení mezery na nový řádek), jinak byla délka čísel v pořádku.
Analýza množiny A trvala 5hodin a 10 minut, Analýza množiny B trvala 7 hodin a 5 minut

test_name	ntup	tsamples	psamples	p-value A	Assessment A	p-value B	Assessment B
diehard_birthdays	0	100	100	0.92147554	PASSED	0.27450955	PASSED
diehard_operm5	0	1000000	100	0.00000000	FAILED	0.00017248	WEAK
diehard_rank_32x32	0	40000	100	0.21656629	PASSED	0.02911547	PASSED
diehard_rank_6x8	0	100000	100	0.09889171	PASSED	0.00013391	WEAK
diehard_bitstream	0	2097152	100	0.99757749	WEAK	0.08469590	PASSED
diehard_opso	0	2097152	100	0.00000000	FAILED	0.00000000	FAILED
diehard_oqso	0	2097152	100	0.10129135	PASSED	0.00094371	WEAK
diehard_dna	0	2097152	100	0.01924592	PASSED	0.10336967	PASSED
diehard_count_1s_str	0	256000	100	0.63992805	PASSED	0.19670739	PASSED
diehard_count_1s_byt	0	256000	100	0.84709485	PASSED	0.99826503	WEAK
diehard_parking_lot	0	12000	100	0.14615348	PASSED	0.48712953	PASSED
diehard_2dsphere	2	8000	100	0.25168448	PASSED	0.41281836	PASSED
diehard_3dsphere	3	4000	100	0.64013216	PASSED	0.94177610	PASSED
diehard_squeeze	0	100000	100	0.00000000	FAILED	0.00000000	FAILED
diehard_sums	0	100	100	0.00504053	PASSED	0.58579000	PASSED
diehard_runs	0	100000	100	0.26412050	PASSED	0.17151654	PASSED
diehard_runs	0	100000	100	0.09677491	PASSED	0.32800650	PASSED
diehard_craps	0	200000	100	0.00000000	FAILED	0.00000000	FAILED
diehard_craps	0	200000	100	0.00000000	FAILED	0.00000000	FAILED
marsaglia_tsang_gcd	0	10000000	100	0.00000000	FAILED	0.00000000	FAILED
marsaglia_tsang_gcd	0	10000000	100	0.00000000	FAILED	0.00000000	FAILED
sts_monobit	1	100000	100	0.33150034	PASSED	0.02724982	PASSED
sts_runs	2	100000	100	0.98571535	PASSED	0.29048340	PASSED
sts_serial	1	100000	100	0.19561699	PASSED	0.01953259	PASSED
sts_serial	2	100000	100	0.02613244	PASSED	0.84787634	PASSED
sts_serial	3	100000	100	0.41490236	PASSED	0.10399995	PASSED
sts_serial	3	100000	100	0.29987448	PASSED	0.01230726	PASSED
sts_serial	4	100000	100	0.43532482	PASSED	0.08174597	PASSED
sts_serial	4	100000	100	0.03841719	PASSED	0.18078597	PASSED
sts_serial	5	100000	100	0.00011255	WEAK	0.00007211	WEAK
sts_serial	5	100000	100	0.01406491	PASSED	0.00000027	FAILED
sts_serial	6	100000	100	0.00032240	WEAK	0.00002066	WEAK
sts_serial	6	100000	100	0.75595446	PASSED	0.00105339	WEAK
sts_serial	7	100000	100	0.16154872	PASSED	0.00001990	WEAK
sts_serial	7	100000	100	0.30869981	PASSED	0.92215355	PASSED
sts_serial	8	100000	100	0.09759497	PASSED	0.00634774	PASSED
sts_serial	8	100000	100	0.55248260	PASSED	0.33621405	PASSED
sts_serial	9	100000	100	0.54238391	PASSED	0.29038149	PASSED
sts_serial	9	100000	100	0.21024461	PASSED	0.05503963	PASSED
sts_serial	10	100000	100	0.03305507	PASSED	0.56041695	PASSED
sts_serial	10	100000	100	0.07993301	PASSED	0.00864060	PASSED
sts_serial	11	100000	100	0.36001851	PASSED	0.89808222	PASSED
sts_serial	11	100000	100	0.04329310	PASSED	0.29633514	PASSED
sts_serial	12	100000	100	0.02553918	PASSED	0.61101293	PASSED
sts_serial	12	100000	100	0.13311270	PASSED	0.05035934	PASSED
sts_serial	13	100000	100	0.42244600	PASSED	0.07884566	PASSED
sts_serial	13	100000	100	0.00298120	WEAK	0.00791281	PASSED
sts_serial	14	100000	100	0.11382237	PASSED	0.16307393	PASSED
sts_serial	14	100000	100	0.41330076	PASSED	0.33082106	PASSED
sts_serial	15	100000	100	0.30801310	PASSED	0.00111609	WEAK
sts_serial	15	100000	100	0.69127286	PASSED	0.71820994	PASSED
sts_serial	16	100000	100	0.52486871	PASSED	0.06792482	PASSED
sts_serial	16	100000	100	0.39235492	PASSED	0.66694056	PASSED
rgb_bitdist	1	100000	100	0.78129016	PASSED	0.02453174	PASSED
rgb_bitdist	2	100000	100	0.84185282	PASSED	0.46436623	PASSED
rgb_bitdist	3	100000	100	0.23864458	PASSED	0.07376870	PASSED
rgb_bitdist	4	100000	100	0.21442101	PASSED	0.67602972	PASSED
rgb_bitdist	5	100000	100	0.69013220	PASSED	0.78345056	PASSED
rgb_bitdist	6	100000	100	0.71135580	PASSED	0.97136258	PASSED
rgb_bitdist	7	100000	100	0.65800481	PASSED	0.68393026	PASSED
rgb_bitdist	8	100000	100	0.47688452	PASSED	0.68515976	PASSED
rgb_bitdist	9	100000	100	0.69302444	PASSED	0.40459667	PASSED
rgb_bitdist	10	100000	100	0.08665845	PASSED	0.28906927	PASSED
rgb_bitdist	11	100000	100	0.55870264	PASSED	0.97718494	PASSED
rgb_bitdist	12	100000	100	0.01044058	PASSED	0.46873469	PASSED
rgb_minimum_distance	2	10000	1000	0.02079940	PASSED	0.00087730	WEAK
rgb_minimum_distance	3	10000	1000	0.00067734	WEAK	0.25829049	PASSED
rgb_minimum_distance	4	10000	1000	0.00020525	WEAK	0.00069560	WEAK
rgb_minimum_distance	5	10000	1000	0.00060793	WEAK	0.00014993	WEAK
rgb_permutations	2	100000	100	0.73524528	PASSED	0.65664643	PASSED
rgb_permutations	3	100000	100	0.05075967	PASSED	0.88608619	PASSED
rgb_permutations	4	100000	100	0.20688088	PASSED	0.04667639	PASSED
rgb_permutations	5	100000	100	0.79143402	PASSED	0.00002300	WEAK
rgb_lagged_sum	0	1000000	100	0.00000003	FAILED	0.00000000	FAILED
rgb_lagged_sum	1	1000000	100	0.00000000	FAILED	0.00000000	FAILED
rgb_lagged_sum	2	1000000	100	0.00000000	FAILED	0.00000000	FAILED
rgb_lagged_sum	3	1000000	100	0.00000000	FAILED	0.00000000	FAILED
rgb_lagged_sum	4	1000000	100	0.00000029	FAILED	0.00002865	WEAK
rgb_lagged_sum	5	1000000	100	0.00000000	FAILED	0.00000000	FAILED
rgb_lagged_sum	6	1000000	100	0.00000003	FAILED	0.04753670	PASSED
rgb_lagged_sum	7	1000000	100	0.00000006	FAILED	0.00000000	FAILED
rgb_lagged_sum	8	1000000	100	0.00000000	FAILED	0.00000262	WEAK
rgb_lagged_sum	9	1000000	100	0.00000001	FAILED	0.00000000	FAILED

```
      rgb_lagged_sum|  10|   1000000|     100|0.00002495|   WEAK          |0.00000475|   WEAK
      rgb_lagged_sum|  11|   1000000|     100|0.00000000|  FAILED         |0.00000000|  FAILED
      rgb_lagged_sum|  12|   1000000|     100|0.00000000|  FAILED         |0.00000000|  FAILED
      rgb_lagged_sum|  13|   1000000|     100|0.00000000|  FAILED         |0.00000000|  FAILED
      rgb_lagged_sum|  14|   1000000|     100|0.00000000|  FAILED         |0.10472120|  PASSED
      rgb_lagged_sum|  15|   1000000|     100|0.00000000|  FAILED         |0.00000000|  FAILED
      rgb_lagged_sum|  16|   1000000|     100|0.00000000|  FAILED         |0.00000143|   WEAK
      rgb_lagged_sum|  17|   1000000|     100|0.00000000|  FAILED         |0.00000000|  FAILED
      rgb_lagged_sum|  18|   1000000|     100|0.00000000|  FAILED         |0.00054439|   WEAK
      rgb_lagged_sum|  19|   1000000|     100|0.00000000|  FAILED         |0.00000000|  FAILED
      rgb_lagged_sum|  20|   1000000|     100|0.00000000|  FAILED         |0.05982638|  PASSED
      rgb_lagged_sum|  21|   1000000|     100|0.00000003|  FAILED         |0.00000000|  FAILED
      rgb_lagged_sum|  22|   1000000|     100|0.00000000|  FAILED         |0.00000000|  FAILED
      rgb_lagged_sum|  23|   1000000|     100|0.00000000|  FAILED         |0.00000000|  FAILED
      rgb_lagged_sum|  24|   1000000|     100|0.00149692|   WEAK          |0.00031115|   WEAK
      rgb_lagged_sum|  25|   1000000|     100|0.00000019|  FAILED         |0.00000000|  FAILED
      rgb_lagged_sum|  26|   1000000|     100|0.00000000|  FAILED         |0.00000000|  FAILED
      rgb_lagged_sum|  27|   1000000|     100|0.00000000|  FAILED         |0.00000000|  FAILED
      rgb_lagged_sum|  28|   1000000|     100|0.00000000|  FAILED         |0.00025996|   WEAK
      rgb_lagged_sum|  29|   1000000|     100|0.00000000|  FAILED         |0.00000000|  FAILED
      rgb_lagged_sum|  30|   1000000|     100|0.00000004|  FAILED         |0.00237523|   WEAK
      rgb_lagged_sum|  31|   1000000|     100|0.00007123|   WEAK          |0.00000000|  FAILED
      rgb_lagged_sum|  32|   1000000|     100|0.00000000|  FAILED         |0.00219766|   WEAK
     rgb_kstest_test|   0|     10000|    1000|0.57890008|  PASSED         |0.07215684|  PASSED
     dab_bytedistrib|   0|  51200000|       1|0.00000000|  FAILED         |0.00000000|  FAILED
            dab_dct| 256|     50000|       1|0.02178273|  PASSED         |0.32586188|  PASSED
Preparing to run test 207.  ntuple = 0
        dab_filltree|  32|  15000000|       1|0.00000000|  FAILED         |0.00000000|  FAILED
        dab_filltree|  32|  15000000|       1|0.00000000|  FAILED         |0.00000000|  FAILED
Preparing to run test 208.  ntuple = 0
       dab_filltree2|   0|   5000000|       1|0.00000000|  FAILED         |0.00000000|  FAILED
       dab_filltree2|   1|   5000000|       1|0.00000000|  FAILED         |0.00000000|  FAILED
Preparing to run test 209.  ntuple = 0
        dab_monobit2|  12|  65000000|       1|1.00000000|  FAILED         |1.00000000|  FAILED
```

Ve výstupu programu je zřejmé že množina B uspěla ve více testech než množina A. Konkrétně si vedla lépe v 8 případech. To mě utvrzuje v názoru že můj algoritmus pomohl "náhodnosti"čipu RNGA. Generátor bude při každém spuštění generovat jiná data, proto je možné že se objeví situace kdy i perfektní, opravdu náhodný generátor neprojde některými testy, ve většině případů je uváděna tolerance 5%[2]

4.2 Vyjádření

Generátor náhodných čísel na FitKitu3 mi přijde rychlý, s dobrou základní funkčností. Osobně si myslím že hodnoty jsou v celku rovnoměrně rozložené a je pro vývojáře pohodlné nemuset implementovat generátor náhodných čísel softwarově, ale s pomocí oscilátorů. Pokud by byla data potřebná na složité simulace či výherní automaty, vybral bych jiný způsob generování dat, jinak pro použití ve hrách a nenáročných aplikacích jsou data dostačující.

Reference

[1] K60 Sub-Family Reference Manual
Document Number: K60P144M100SF2V2RM
Freescale Semiconductor, Inc

[2] dieharder (1) - Linux Man Pages
https://www.systutorials.com/docs/linux/man/1-dieharder/