MCSL2017

Lab2 Optional Problem Notes (+10%)

TA:柴俊瑜

cychai2.cs06g@nctu.edu.tw

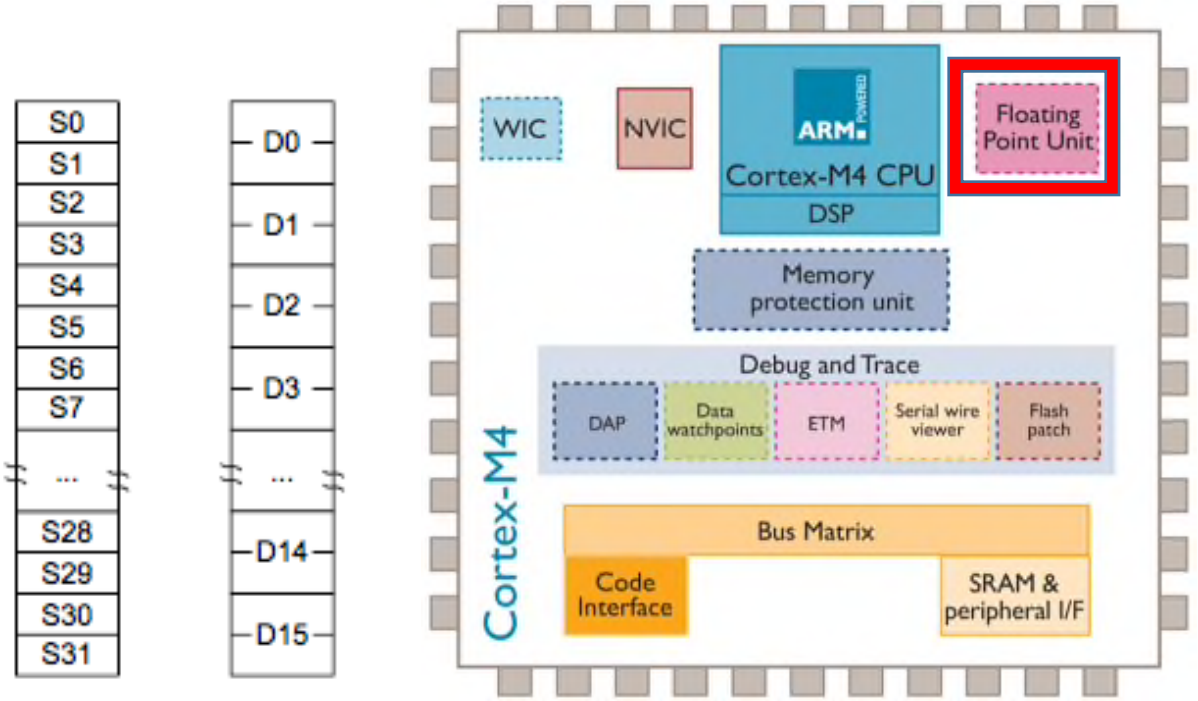
Lab 2.4 Monte Carlo Method for estimating Pi(**Optional**)

You will learn…

- Enabling **FPU**
  - Check CH4.6.6 in **M4 Programming Manual** (Find the code!)
- Using Floating Point Instructions
  - Check CH3.10 in **M4 Programming Manual**
- Enabling the **RNG**(Random Number Generator) hardware on STM32
  - Check CH24, CH24.3.1 in **STM32L4x6 Reference manual** to enable the RNG by simply setting the RNGEN bit to 1
  - We don't use interrupt mode, IE bit should be 0
- Get the random number from RNG and convert to Float representation
  - RNG generates 32bit random number(can be viewed as an Unsigned Integer U32)
  - You may need to use VCVT.F32.U32to convert a loaded U32 to Float representation
- Estimate the Pi using Monte Carlo Method
  - Sample 1000000 points (~20 seconds in TA's Implementation)
  - Calculate Pi (should be around 3.14X in TA's Implementation (float32))

2.4.1 FPU and Float point manipulation

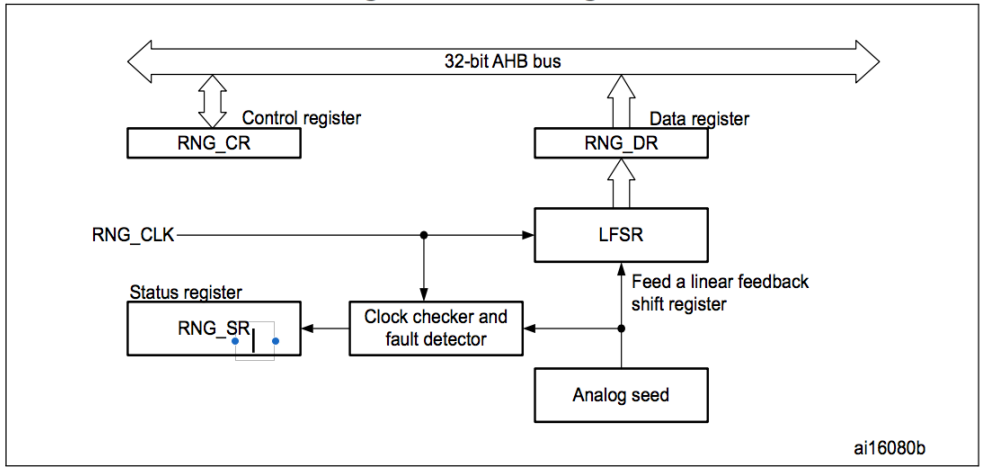
- FPU is a dedicated HW for float/double calculation
- FPU has its own register
  - S0-S31 (32bit) for Float32
  - Also viewed as D0-D15 for Double64
  - It's own FPSCR similar to APSCR
- Move data between R0-R31 ⇔S0-S31
  - VMOV Sn,Rn
- Load data from some address
  - VLDR.F32 Sn,[Rn] //Rn with address
- Load Immediate to S0-S31
  - VMOV.F32 Sn, #Immediate_float32
- Floating Point Manipulation
  - VADD, VSUB, VMUL, VDIV, VABS……
  - VCMP
    - with VMRS APSR_nzcv,FPSCR
    - Copy the FPSCR to APSCR, then you can use the branch instructions like bgt,beq,blt…



2.4.2 RNG (True) Random Number Generator

- Random seed from **analog signal noise**
- Generated 32bit Random number in RNG_DR register
- RNG related clock settings is provided in the code template, you only need to set the RNGEN bit to 1
- Map the random U32 values to the range you need !

Figure 170. Block diagram



2.4.2 RNG (True) Random Number Generator

24.4.1 RNG control register (RNG_CR)

Address offset: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IE	RNGEN	Res.	Res.
												rw	rw		

Bits 31:4 Reserved, must be kept at reset value

Bit 3 **IE:** Interrupt enable

0: RNG Interrupt is disabled
1: RNG Interrupt is enabled. An interrupt is pending as soon as DRDY=1 or SEIS=1 or CEIS=1 in the RNG_SR register.

Bit 2 **RNGEN:** Random number generator enable

0: Random number generator is disabled
1: random Number Generator is enabled.

Bits 1:0 Reserved, must be kept at reset value

2.4.2 RNG (True) Random Number Generator

24.4.2 RNG status register (RNG_SR)

Address offset: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SEIS	CEIS	Res.	Res.	SECS	CECS	DRDY
									rc_w0	rc_w0			r	r	r

Bit 0 **DRDY:** Data ready

0: The RNG_DR register is not yet valid, no random data is available

1: The RNG_DR register contains valid random data

Note: An interrupt is pending if IE = 1 in the RNG_CR register.

Once the RNG_DR register has been read, this bit returns to 0 until a new valid value is computed.

2.4.2 RNG (True) Random Number Generator

24.4.3 RNG data register (RNG_DR)

Address offset: 0x08

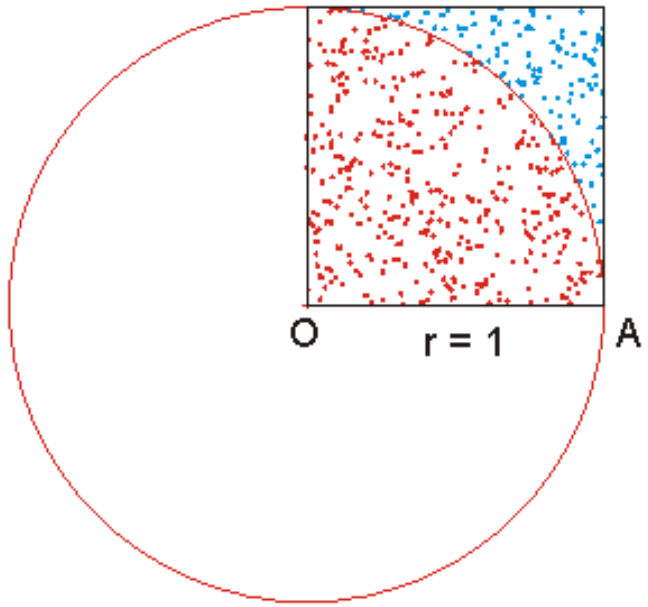
Reset value: 0x0000 0000

The RNG_DR register is a read-only register that delivers a 32-bit random value when read. After being read, this register delivers a new random value after a maximum time of 40 periods of the RNG_CLK clock. The software must check that the DRDY bit is set before reading the RNDATA value.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RNDATA															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RNDATA															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

2.4.3 Estimation of Pi

- Sample N points in unit area
- M points in the unit circle
- $Estimation\ of\ \pi = \frac{4*M}{N}$
- Around 3.14X for N=1 Million



(x)= Variables | Breakpoints | Expressions | Registers | I/O Registers

Name	Value
faultmask	0
control	0
s0	3.14456797
s1	1000000

control	0
s0	3.14135003
s1	10000000