Errata of

Embedded Systems with ARM Cortex-M3 Microcontrollers in Assembly Language and C

ISBN-10: 0982692625

Yifeng Zhu

For the printing dated August 5, 2014

March 23, 2015

Thank you all for providing me feedbacks and corrections!

Chapter 1. See a Program Running

- Page 6: “we can allocate a 4KB region for the data memory and a 156KB region for the instruction memory, as shown in Figure 1-6” should read as “we can allocate a 4KB region for the instruction memory and a 256KB region for the data memory, as shown in Figure 1-6.”
- Page 10, Figure 1-9, the 0x00000000-0x08000000 range is 128 MB, not 126 MB.
- On Page 16, in Tab 1-2, add the last entry (hex 0x2000) and (binary 0010000000000000) in column "Machine Program"
- On Page 18, in the last sentence, “takes four **types** in memory" should be "bytes", not “types”

Chapter 2. Data Representation

- Page 35: “Carry = Not Barrow” should be “Carry = Not Borrow”
- Page 41: With the text box on the right, it should be “A processor sets up both the carry flag and overflow flag.”
- Page 45, “`0b11101` = 3” should be “`0b11101` = -3”
- Page 48, at the end of the first paragraph, "The ASCII value of the ZERO character is **0x50**" should read as "The ASCII value of the ZERO character is **0x30**".
- Page 48, Table 2—7, the letter of ASCII value 0x72 and 0x74 should be ‘r’ and ‘t’, respectively.
- Page 49, the character comparison order listed in the first paragraph '9' > '0' > '1' should be '9' > '1' > '0’.
- Page 51, 2.6 Exercises, Question 2, “Assume a five-bit system” should be “Assume a six-bit system”

Chapter 3. ARM Instruction Set Architecture

- Page 57, in Figure 3-3, “x DCW **-1**” should be “x DCW **-2**”.
- Page 64, in the table, the description of SMULL and UMULL are incorrect.
  SMULL (signed long multiply)***,***
  SMLAL (signed long multiply, with accumulate),
  UMULL (unsigned long multiply)***,***
  UMLAL (unsigned long multiply, with subtract)
- Page 64, in the middle table, the explanation of REVSH should be “reverse byte order in the bottom halfword, and sign extend to 32 bits.”
- Page 65, in the table, "LTRH" should be "LDRH", and all "SDxx" should be "STxx"

Chapter 4. Arithmetic and Logic

- Page 74: "**RRX** (rotate right with extend) works similarly to ROR except the carry bit joins the rotate circle. Similarly, the carry bit may be used to update the carry flag." This statement is incorrect. The correction is "**RRX** (rotate right with extend) works similarly to ROR except that the carry bit joins the rotate circle and RRX rotates the data by only one bit"
- Page 77, "`r0 = r0 + r0 << 3 = 10 × r0`" should be "`r0 = r0 + r0 << 3 = 9 × r0`".
- Page 80, "`r1 = r0 + r0 >> 3 = r0 + r0/9 (unsigned)`" should be "`r1 = r0 + r0 >> 3 = r0 + r0/8 (unsigned)`".
- Page 80, "`r1 = r0 + r0 >> 3 = r0 + r0/9 (signed)`" should be "`r1 = r0 + r0 >> 3 = r0 + r0/8 (signed)`".
- Page 80,

```
ADD r1, r0, r0, LSL #3  ⇔  MOV r2, #11      ; r2 = 11
                           MUL r1, r0, r2   ; r1 = r0 * 11
should be

ADD r1, r0, r0, LSL #3  ⇔  MOV r2, #9       ; r2 = 9
                           MUL r1, r0, r2   ; r1 = r0 * 9
```

- Page 86, In Example 4-4, "signed int_32 c; // a signed 21-bit integer" should be "signed int_32 c; // a signed 32-bit integer"
- Page 86, "-2" is 0xFFFE, not 0xFFFF in Example 4-4.
- Page 87, switch TEQ and TST in Table 4-9.

CMP Rn, Op2	Compare	Set flags on Rn – Op2
CMN Rn, Op2	Compare negative	Set flags on Rn + Op2
TST Rn, Op2	Test	Set flags on Rn AND Op2
TEQ Rn, Op2	Test equivalence	Set flags on Rn EOR Op2

- Page 87, "Note **TEQ** instruction cannot check the equivalence of two operands" Should be "Note **TST** instruction cannot check the equivalence of two operands".

Chapter 5. Load and Store

- On page 95, at the beginning "A load instruction is used to" => "used" missing
- Page 100, two questions are numbered as "3". The last one should be numbered as "4".

Chapter 6. Branch and Conditional Execution

- On Page 107, at the end of the third paragraph, “ADDED” should be “ADDEQ”.
- On Page 109, Example 6-5

C Program	Assembly Program

`// x is a signed integer`
`if(x <= 20 || x >= 25){`
`  a = 1`
`}`
	`  ; r0 = x`
	`  CMP  r0, #20  ; compare x and 20`
	`  BGT  endif  ; go to endif if x > 20`
	`  CMP  r0, #25  ; compare x and 25`
	`  BLT  endif  ; go to endif if x < 25`
	`then  MOV  r1, #1  ; a = 1`
	`endif`

The assembly program should be

C Program	Assembly Program

`// x is a signed integer`
`if(x <= 20 || x >= 25){`
`  a = 1`
`}`
	`  ; r0 = x`
	`  CMP  r0, #20  ; compare x and 20`
	`  BLE  then  ; go to then if x ≤ 20`
	`  CMP  r0, #25  ; compare x and 25`
	`  BLT  endif  ; go to endif if x < 25`
	`then  MOV  r1, #1  ; a = 1`
	`endif`

- Page 111, Section 6.5. In the flow chart, the ***then*** clause should be “b=3” and the ***else*** clause should
  be “b = 4”.
- Page 116, Section 6.10, “`LDRB r2, [r1]`” should be “`LDRB r2, [r0]`”

Chapter 7. Structured Programming

Chapter 8. Subroutines

Chapter 9. 64-bit Data Processing

Chapter 10. Mixing C and Assembly

- Page 218, Exercise Question 4,

Textbook	Correction

`char * search (char * s, char c) {`
`  char *p = NULL;`
`  for(; *s; s++)`
`  p = s;`
`  return p;`
`}`
	`char * search (char * s, char c) {`
	`  char *p = NULL;`
	`  for(; *s; s++)`
	`  if (*s == c)`
	`  p = s;`
	`  return p;`
	`}`

Chapter 11. Fixed-point and Floating-point Arithmetic

- Page 239, Table 11-2, the rounding up result of -0.123456 should be -0.12345, instead of -0.12344.
- Page 240, Table 11-3, "3. If b_{m+1} = 1 and b_S = 1" should be "3. If b_{m+1} = 1 and b_S = 0".

Chapter 12. Interrupt

- Page 254, Table 12-1, "Softer trigger bit" should be "Software trigger bit"
- Page 255, at the end of the page, "enable the ADC1 interrupt" should be "enable the Timer 7 interrupt"
- Page 256, Example 12-1,
  "`LSR  r2,r0,#3  ; Memory offset (in bytes): IRQn >> 3`"
  should be
  "`LSR  r2,r0,#5  ; Memory offset (in bytes): IRQn >> 5`"
- Page 257,

  "we only shift right IRQn by three bits instead of five bits This is because the register array index is based on words in the above in C code. However, in the assembly code, the STR instruction is based on the memory address, which is always in terms of bytes.

$$\boldsymbol{Memory\ Address\ Offets\ (in\ bytes) = Interrupt\ Number\ \div 8}$$

  "

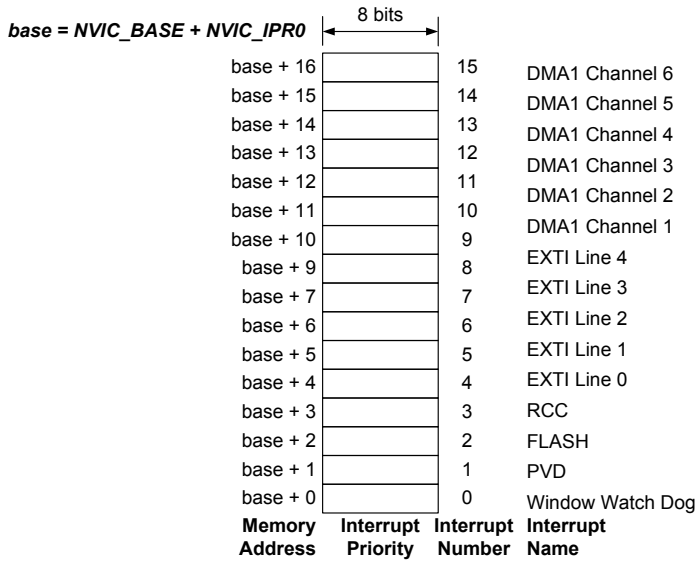
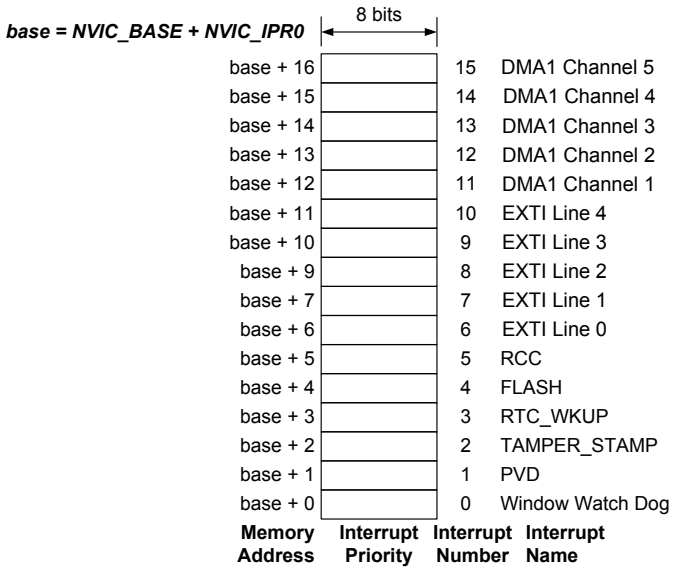
  should be

  "we shift right IRQn by five bits. This is because the register array index is based on words in the above in C code and the STR instruction is based on the memory address, which is always in terms of bytes.

$$\boldsymbol{Memory\ Address\ Offets\ (in\ bytes) = Interrupt\ Number\ \div 32}$$

  "

- Page 255, “STM32L has only 44 peripheral interrupts” should read as “STM32L has only 45 peripheral interrupts”
- Page 259, Figure 12-8

Textbook	Correction
	

- Page 259, “For example, the following code changes the priority of EXTI Line 0, whose interrupt number is 4, to the lowest priority.” should be corrected as “whose interrupt number is 6”.
- Page 259, “`NVIC->IP[4] = 0xF0;`” should read as “`NVIC->IP[6] = 0xF0;`”.
- Page 263, in Figure 12-9 Caption, “**HIS**” should be “**HSI**”.

Chapter 13. Instruction Encoding and Decoding

Chapter 14. Generic-purpose I/O

- Page 294, in Figure 14.1, the “output pin” should be “input pin.” The “output” should be “input”. The figure title should be “The input pin is pulled up internally”.
- Page 294, in Figure 14.2, the “output pin” should be “input pin.” The “output” should be “input”. The figure title should be “The input pin is pulled down internally”.
- Page 298 on the third paragraph of 14.4, "higher slow rate" should be "higher slew rate".
- Page 302, in the 3rd bullet, “push-pall” should be “push-pull”
- Page 307, in the example program, “10 Mhz (01)” should be “10 Mhz (10)”
- Page 311, in the first paragraph, “a row pin is set to one” should be “a row pin is set to zero”. In the same paragraph, “its output value is one” should be its output value is zero”.
- Page 319, in the example program, “10 Mhz (01)” should be “10 Mhz (10)”

Chapter 15. General-purpose Timers

- Page 325, “In this example, the prescaler factor is set as 15” should be “In this example, the prescaler factor is set as 63”.
- Page 325, We set the ARR register as **199**. Therefore, the timer generates a pulse in each period of

$$T_{PWM} = \frac{ARR + 1}{f_{CK\_CNT}} = \frac{200}{2^{15}Hz} = 6.1\ ms$$

- Page 333, Figure 15-14 is repeated.
- Page 325, In PWM mode 1, we have

$$duty\ cycle\ = \frac{CCR}{ARR} \times 100\% \qquad \text{Correction: } duty\ cycle\ = \frac{CCR}{ARR+1} \times 100\%$$

  In PWM mode 2, we have

$$duty\ cycle\ = (1 - \frac{CCR}{ARR}) \times 100\% \qquad \text{Correction: } duty\ cycle\ = (1 - \frac{CCR}{ARR+1}) \times 100\%$$

- Page 336, Example 15-6,
  “ NVIC_EnableIRQ(TIM4_IRQn); // Enable EXTI0_1 interrupt in NVIC”
  should be
- “ NVIC_EnableIRQ(TIM4_IRQn); // Enable Timer 4 interrupt in NVIC” Page 341, “: the PWM output is high if the counter is larger than the content of CCR.” Should be “: the PWM output is high if the counter is larger than or equal to the content of CCR.”

Chapter 16. Stepper Motor Control

- Page 350, In Example 16-3, “unsigned char FullStep[4] = {0x9, 0x8, 0xa, 0x2, 0x6, 0x4, 0x5, 0x1};” should be “unsigned char FullStep[8] = {0x9, 0x8, 0xa, 0x2, 0x6, 0x4, 0x5, 0x1};”

Chapter 17. Liquid-crystal Display (LCD)

- Page 356, second paragraph, “The voltage across segment lines 1, 2, **3,** 4, 5 and 7 has” should be “The voltage across segment lines 1, 2, 4, 5, 7 and **8** has”
- Page 365, in the flow chart (Figure 17-7), the LCD configuration function ***LCD_Configure*** needs to set up the pulse on duration to eliminate the disappearance of some segments.
  // Set Pulse ON duration
  // Use high drive internal booster to provide large drive current
  // Set the duration that the low-resister voltage divider is used
  LCD->FCR |= 0x7 << 4; // PON[2:0] = 0x111
- On Page 365, in the flow chart (Figure 17-7), “Wait until the LCD is enabled by checking LCDEN bit bit of LCD_CR “ should be corrected as “Wait until the LCD is enabled by checking the ENS bit of LCD_SR”.

- On Page 366, in the 'A' and '2' examples, the fifth row should be "Q K Col P", not "Q K Q P"

Chapter 18. Real-time Clock (RTC)

Chapter 19. Direct Memory Access (DMA)

Chapter 20. Analog-to-Digital Converter

- Page 390, "For example, when the switch is closed for a time period of $\mathbf{2}T_c$, the voltage across the capacitor V_C is only 95.02% of the input voltage V_{in} " should read as "For example, when the switch is closed for a time period of $\mathbf{3}T_c$, the voltage across the capacitor V_C is only 95.02% of the input voltage V_{in}"

- Page 395,

$$V = \frac{Digital\ Value}{2^n} \times V_{REF}$$

should be

$$V = \frac{Digital\ Value}{2^n - 1} \times V_{REF}$$

- Page 407, in Figure 21-1, we should remove the ground. Vref cannot be shorted to the ground.

Chapter 21. Digital-to-Analog Converter

Chapter 22. Serial Communication Protocols

Chapter 23. Multitasking

Appendix A: Cortex-M3 16-bit Thumb-2 Instruction Encoding

Appendix B: Cortex-M3 32-bit Thumb-2 Instruction Encoding

Appendix C: HID Codes of a Keyboard

Bibliography

Index