Chapter 1. See a Program Running

- Page 22, bullet list #2, the binary code for memory address 0x08000162 is 0x680A, not 0x680.
- Page 22, “This instruction loads the value of variable a into register r1.” It should be r2.

Chapter 2. Data Representation
Chapter 3. ARM Instruction Set Architecture
Chapter 4. Arithmetic and Logic
Chapter 5. Load and Store
Chapter 6. Branch and Conditional Execution
Chapter 7. Structured Programming
Chapter 8. Subroutines
Chapter 9. 64-bit Data Processing
Chapter 10. Mixing C and Assembly
Chapter 11. Interrupt
Chapter 12. Fixed-point and Floating-point Arithmetic
Chapter 13. Instruction Encoding and Decoding

Chapter 14. Generic-purpose I/O

- On Page 355, the demo code given in the middle
  
  ORR r1, r1, #(1<<6) ; Set bit 6
  should be
  ORR r1, r1, #(1<<2) ; Set bit 2

- On Page 363, Example 14-6,

<table>
<thead>
<tr>
<th>Incorrect code</th>
<th>Correct code</th>
</tr>
</thead>
<tbody>
<tr>
<td>void TIM4_IRQHandler(void) {</td>
<td>void TIM4_IRQHandler(void) {</td>
</tr>
</tbody>
</table>
|   ...                                                                       |   ...
|   if((GPIOA->IDR & 0x1) == 0x1){ // check                                  |   if((GPIOA->IDR & 0x1) == 0x1){ // check                                  |
|     counter++; // button is pressed                                         |     counter++; // button is pressed                                         |
|     if (counter >= 4) {                                                     |       if (counter >= 4) {                                                   |
|       pressed = 1; // set the flag                                          |         pressed = 1; // set the flag                                         |
|       counter = 0; // reset counter                                        |         counter = 0; // reset counter                                        |
|     } else { // button is not pressed                                       |       } else { // button is not pressed                                     |
|       counter = 0; // reset counter                                        |         counter = 0; // reset counter                                        |
| }                                                                            | }                                                                             |
Chapter 15. General-purpose Timers
Chapter 16. Stepper Motor Control
Chapter 17. Liquid-crystal Display (LCD)
  • Page 440, caption of Table 17-2, “encoding of five letters (A-Z)” should be “encoding of five letters (A-E)”.
Chapter 18. Real-time Clock (RTC)
Chapter 19. Direct Memory Access (DMA)
Chapter 20. Analog-to-Digital Converter
  • Page 265, Example 11-13, “EXTI->FTSR &= ~EXTI_FTSR_RT3;” should be EXTI->FTSR &= ~EXTI_FTSR_FT3;

Chapter 21. Digital-to-Analog Converter
Chapter 22. Serial Communication Protocols
  • Page 576, in Example 22-27, Send data to an SPI slave
    1. SPIx->DR = txBuffer[i];
       should be: *((volatile uint8_t*)&SPIx->DR) = txBuffer[i];
    2. rxBuffer[i] = SPIx->DR;
       should be: rxBuffer[i] = *((volatile uint8_t*)&SPIx->DR);

  • Page 577, in Example 22-28, Receive data from an SPI slave
    1. SPIx->DR = 0xFF;  // A dummy byte
       should be: should be: *((volatile uint8_t*)&SPIx->DR) = 0xFF
    2. rxBuffer[i] = SPIx->DR;
       should be: rxBuffer[i] = *((volatile uint8_t*)&SPIx->DR);

Chapter 23. Multitasking
  • Page 405 and 406, run the pseudo instruction “CPSID I”
  the pseudo instruction “CPSIE I”

Chapter 24. Digital Signal Processing