Errata for Fundamentals of Embedded Software (2nd edition)

Note: Some of these may already have been incorporated into earlier printings

Chapter 2:

Page 17, 2nd paragraph: Change every occurrence of 0.231 to 0.221

Answers to Selected Problems

Problem 3a. Answer is -13 (not -115)

Chapter 3:

Page 40, Table in middle of page:

First decimal constant should be 74 instead of 9

30 = 2^5A – 2^1A instead of 2^6A – 2^1A.

Answers to Selected Problems

Answer given for Problem 3 is really for Problem 3e

Answer given for Problem 7 is really for Problem 7a

Answer given for Problem 8 is really for Problem 8c

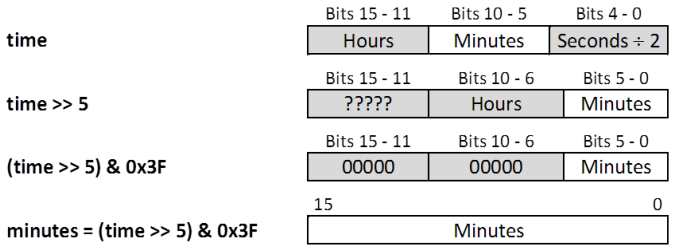
Chapter 4:

Page 60: Replace Table 4-7 with the following table:

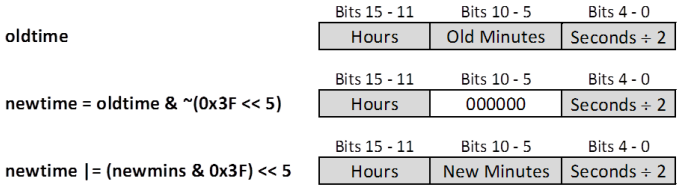
TABLE 4–7 Interpreting the Bitwise-XOR

a	*b*	*a* XOR *b*	Interpretation
0	0	0	If control bit *a* is 0, bit *b* propagates through to the result unchanged.
	1	1 (*b*)	
1	0	1	If control bit *a* is 1, bit *b* propagates through to the result inverted.
	1	0 (*b*')	

Page 62: Replace Figure 4-5 with the following figure:



Page 63: Replace Figure 4-6 with the following figure:



Page 77, problem 11, part l: Change !0(1) to !0

Page 78, problem 17: Change the uppercase A in bit position 7 to a lowercase a.

Page 79, problem 23: Change BYTE8 to uint_8.

Page 79, problem 25: The left-most number above the rectangles should be 31, not 1

Chapter 5:

Page 91, 1st paragraph.

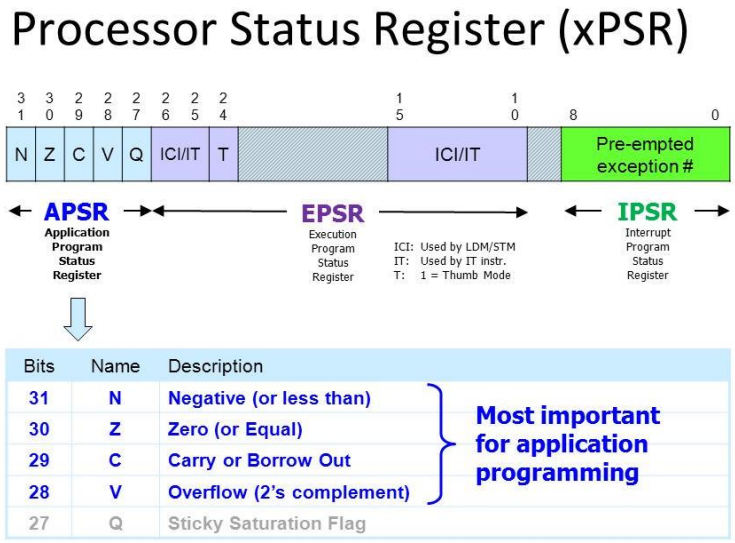
Change the following sentence from:

The EPSR holds the exception number during exception processing.

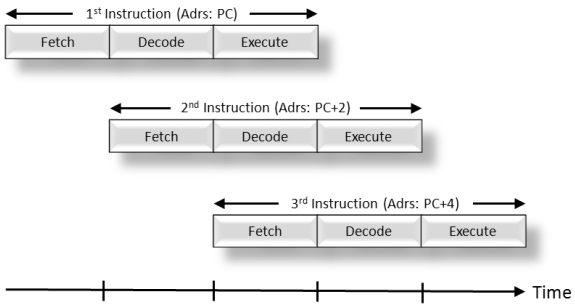
To:

The IPSR holds the pre-empted exception number during exception processing.

Page 92: Replace Figure 5-12 with the following figure:



Page 93: Replace Figure 5-14 with the following figure:



Page 99, problem 12, part (a): Change "whoose" to "whose".

Page 100, problem 19, part (d): Change "PSW" to "PSR"

Answers to Selected Problems

Problem 27. Answer is B & D (not just B)

Chapter 6:

Page 105, Table 6-1, last row: Change Notes to "<mem> may only be [R_n] or [R_n,#imm]"

Page 108: Replace Table 6-4 with the table shown below:

TABLE 6-4 Offset Addressing Options

Syntax	***Memory Address***	***Example***	***Notes***
[R_n]	R_n	[R5]	
[R_n,#constant]	R_n + constant	[R5,#100]	
[R_n,R_m]	R_n + R_m	[R4,R5]	Not available with LDRD or STRD. Only register R_n may be used with these instructions.
[R_n,R_m,LSL #constant]	R_n + (R_m << constant)	[R4,R5,LSL #3]	

Note: "LDR R0,data" is a special PC-relative case of [R_n,#constant]

Page 116, end of section 6.8.3:

Replace the #20 constant in the BIC instruction by #x20 (hex), so that it reads "BIC R0,R0,#x20".

Page 121, problem 6c:

The operands in the MLS instruction should be separated by commas, not periods.

Page 121, problem 7:

Add the following declaration: uint16 *pu16 ;

Page 122, problems 7w and 7x:

Change p16 to pu16.

Chapter 7:

Page 125: Replace Table 7-2 with …

TABLE 7-2 ARM Condition Codes

Code	*Meaning*	*Requirements*
EQ	**EQ**ual	Z = 1
NE	**N**ot **E**qual	Z = 0
HS	Unsigned ≥ ("**H**igher than or **S**ame")	C = 1
LO	Unsigned < ("**LO**wer")	C = 0
HI	Unsigned > ("**HI**gher")	C = 1 && Z = 0
LS	Unsigned ≤ ("**L**ower or **S**ame")	C = 0 || Z = 1
GE	Signed ≥ ("**G**reater than or **E**qual")	N = V
LT	Signed < ("**L**ess **T**han")	N ≠ V
GT	Signed > ("**G**reater **T**han")	Z = 0 && N = V
LE	Signed ≤ ("**L**ess than or **E**qual")	Z = 1 || N ≠ V
CS	**C**arry **S**et *(synonym for HS)*	C = 1
CC	**C**arry **C**lear *(synonym for LO)*	C = 0
MI	**MI**nus/negative	N = 1
PL	**PL**us - positive or zero (non-negative)	N = 0
VS	o**V**erflow **S**et	V = 1
VC	o**V**erflow **C**lear	V = 0
AL	**AL**ways (unconditional)	*(Rarely used)*

Page 125, first paragraph, last sentence: Replace "HI and LS" by "HS, LO, HI and LS".

Page 126, remove third paragraph that begins, "You may have noticed…"

Page 126, fourth paragraph, first sentence: Replace "7-3 and 7-4" by "7-3".

Page 126, Figure 7-3: Change "BG L1 ; NO!" to "BGT L! ; NO!"

Page 127: Remove Figure 7-4.

Page 138, problem 6 (b): Change "void f2(char)" to "void f2(signed char)".

Page 139, problem 6 (l): Change "void swap(long *p1, *p2)" to "void swap(long *p1, long *p2)".

Answers to Selected Problems

Problem 2a. The correct solution is:

```
LDR   R0,x
LDR   R1,y
CMP   R1,R0
BLS   Else
LDR   R0,=6
STR   R0,z
```

```
        B       EndIf
Else:   STR     R0,z
EndIf:
```

Chapter 8:

Page 142, section 8.1.3, numbered paragraphs 1 and 2: Change all occurences of "PSW" to "PSR".

Page 143, first paragraph, last sentence: Change "PSW" to "PSR".

Page 151, Figure 8-11: Change "BZ" to "BEQ"

Page 152, Figure 8-12: Change "BZ" to "BEQ"

Page 157, Problem 10, part (e): Change "PSW" to "PSR"

Chapter 11:

Page 208, problem 11, the numbered point that appears just before the return statement should be labeled "6", not "5".

Problems at end of chapter:

3d: Change to "may cause the same variable to be initialized more than once during execution?"

3f: Change to "destroy objects invisibly during program execution?"

Chapter 12:

Replace Table 12-2 with the following:

TABLE 12–2 When "x = 0" becomes a Critical Section

Architecture	Operand (x)	8-bit CPU	16-bit CPU	32-bit CPU
Load/Store Architecture: The only instructions that can reference memory are Load and Store. E.g., RISC processors, such as ARM and MIPS	8 bits			
	16 bits	*		
	32 bits	**X**	*	
	64 bits	**X**	**X**	*
Other Architectures: Instructions like Add and Subtract may have memory operands. E.g., CISC processors, such as Intel x86.	8 bits			
	16 bits	*		
	32 bits	**X**	*	
	64 bits	**X**	**X**	*

*The table assumes that a single instruction can store a memory operand no larger than the

word size of the processor. However, some processors have instructions that can store a double-length operand (e.g., STRD).

Page 220, Figure 12-12, replace “LORD” in the first column by “LDRD”.