

# Programming the SX Microcontroller by Günther Daubach

## Errata per October 16, 2003

Page	Paragraph	Reads	Should read
11	8 <sup>th</sup>	...two hexadecimal digits <i>instead</i> ...	...two hexadecimal digits...
13	2 <sup>nd</sup>	...SX itself <i>it</i> has...	...SX itself. <i>It</i> has...
49		This blank page was inserted by the printing company after I have created the TOC and the index. Therefore, most of the references in the TOC and the index have an offset of -1.	
51	4 <sup>th</sup>	...written bank \$00 up to bank \$2f...	...written bank \$00 up to bank \$1f...
56	6 <sup>th</sup> last	...Line 5 contains the instruction...	...Line 4 contains the instruction...
56	6 <sup>th</sup> last	...in line 6 is executed...	...in line 5 is executed...
56	last 4	replace the last four paragraphs	<p>At the first "round" through the loop, register 8 is cleared (at line 6), FSR is incremented to \$09 at line 7 and the jump to ClearData is then executed.</p> <p>When the loop starting at ClearData executes the next time, FSR contains \$09 in line 4. As FSR bit 4 is not yet set, line 5 is executed again. FSR bit 3 is already set, so this does not change the contents of FSR at all.</p> <p>This is continued until FSR has reached a value of \$10. From then on, line 5 will be skipped.</p> <p>As soon as FSR's contents reaches \$20, i.e. when bit four is clear, line 5 will be executed again which sets FSR bit three. Again, this skips the "restricted area" that would be mapped into the address space for \$20 to \$27.</p>
66	4 <sup>th</sup>	<p>Replace the code example by the following version:</p> <pre> org          \$08          localTemp0 ds           1  org          \$10          BankA equ         \$  org          \$30          BankB equ         \$  org          \$50          BankC equ         \$  SR1 mov          localTemp0, fsr    ; save the current bank (BankA) bank        BankB             ; fsr now "points" to BankB ; ; some instructions ; call SR2 ; ; some more instructions ; mov          fsr, localTemp0   ; restore the saved bank (should ; be BankA, but actually, it is ; BankB in this example) </pre>	

		<pre> ret SR2   mov     localTemp0, fsr    ; save the current bank (BankB)   bank   BankC              ; fsr now "points" to BankC   ;   ; some instructions   ;   mov     fsr, localTemp0   ; restore the saved bank (BankB) ret Main   bank   BankA   call  SR1                  ; after returning from SR1, fsr                              ; should "point" to BankA again,                              ; but it "points" to BankB in-                              ; stead.   ;   ; etc. </pre>	
67	2 <sup>nd</sup>	<p>Replace the code example by the following version:</p> <pre> org     \$08 localTemp0  ds    1 localTemp1  ds    1  org     \$10 BankA    equ    \$  org     \$30 BankB    equ    \$  org     \$50 BankC    equ    \$  SR1   mov     localTemp0, fsr    ; save the current bank (BankA)   bank   BankB              ; fsr now "points" to BankB   ;   ; some instructions   ;   call  SR2   ;   ; some more instructions   ;   mov     fsr, localTemp0   ; restore the saved bank (BankA) ret SR2   mov     localTemp1, fsr    ; save the current bank (BankB)   bank   BankC              ; fsr now "points" to BankC   ;   ; some instructions   ;   mov     fsr, localTemp1   ; restore the saved bank (BankB) ret Main   bank   BankA   call  SR1                  ; after returning from SR1, fsr                              ; "points" to BankA again   ;   ; etc. </pre>	
70	4th	...The <i>PopFSR</i> subroutine is used...	...The <i>PushFSR</i> subroutine is used...
86	2 <sup>nd</sup> last	Result: <i>10111101</i>	Result: <i>00101000</i>
91	last	...In case bit 7 of the low result byte was set <i>before</i> , the C flag is set now, otherwise, it is clear...	...In case bit 7 of the low result byte was <i>previously</i> set, the C flag is now set, otherwise it is clear...
95	1 <sup>st</sup>	...The Z flag is set if the result is zero, <i>else</i> it is cleared...	...The Z flag is set if the result is zero, <i>otherwise</i> it is cleared...
95	3 <sup>rd</sup>	...The C flag is cleared if the result is negative <i>else</i> , it is set. The DC flag will be cleared on an underflow from the high to the low nibble <i>else</i> , it is set...	...The C flag is cleared if the result is negative <i>otherwise</i> , it is set. The DC flag will be cleared on an underflow from the high to the low nibble

			<i>otherwise</i> , it is set...
95	8 <sup>th</sup>	...The Z flag is set if the result is zero, <i>else</i> it is cleared...	...The Z flag is set if the result is zero, <i>otherwise</i> it is cleared...
95	2 <sup>nd</sup> last	...The Z flag is set if the result is zero, <i>else</i> it is cleared...	...The Z flag is set if the result is zero, <i>otherwise</i> it is cleared...
97	1 <sup>st</sup>	... the Z flag is set <i>else</i> , it is cleared...	... the Z flag is set <i>otherwise</i> , it is cleared...
98	8 <sup>th</sup>	... If <b>fr2</b> contains zero, the Z flag is set <i>else</i> , it is cleared...	...If <b>fr2</b> contains zero, the Z flag is set <i>otherwise</i> , it is cleared...
98	4 <sup>th</sup> last	... If <b>fr</b> contains zero, the Z flag is set <i>else</i> , it is cleared...	... If <b>fr</b> contains zero, the Z flag is set <i>otherwise</i> , it is cleared...
99	2 <sup>nd</sup>	... If <b>fr</b> contains zero, the Z flag is set <i>else</i> , it is cleared...	... If <b>fr</b> contains zero, the Z flag is set <i>otherwise</i> , it is cleared...
104	41 <sup>st</sup> line	<b>mov Button, w</b>	Remove this line
105	3 <sup>rd</sup>	...that is cleared in the main program.	Remove this part of the sentence.
105	3 <sup>rd</sup>	...but <i>well</i> use...	...but <i>we'll</i> use...
105	5 <sup>th</sup>	...we mask out bit 4... ...When bit 4 is not set...	...we mask out bit 3... ...When bit 3 is not set...
105	last	...version of the <b>TimeEater</b> subroutine:	...version of the <b>TimeEater</b> subroutine that replaces the original code in <i>TUT019.src</i> .
108	2 <sup>nd</sup>	...built in special <i>resistors</i> ...	...built-in special <i>registers</i> ... ☺
111	2 <sup>nd</sup>	...the <b>WKPMND_B</b> register...	...the <b>WKPMND_B</b> register...
112	last	...increments the register at \$08...	...increments the register at \$09...
116	2 <sup>nd</sup>	...10ns * 256 * 256 = 1.3ms...	...20ns * 256 * 256 = 1.3ms...
130	2 <sup>nd</sup> last	...if this <i>but</i> is set...	...if this <i>bit</i> is set...
132	2 <sup>nd</sup> last	...that is <i>replaces</i> by the...	...that is <i>replaced</i> by the...
141	7 <sup>th</sup>	...SwapRegs, Val1, Val2...	...SwapRegs Val1, Val2...
143	3 <sup>rd</sup> last	...where the <i>to</i> "real" instructions...	...where the <i>two</i> "real" instructions...
163	2 <sup>nd</sup>	...\$200 jmp <i>Start</i> ...	...\$200 jmp <i>TooFar</i> ...
164	2 <sup>nd</sup> last	...call <i>@Farther</i> ...	...@call <i>Farther</i> ...
167	3 <sup>rd</sup> last	...shows the desired <i>the</i> RC...	...shows the desired RC...
328	3 <sup>rd</sup> last	...again when the clock is running and...	...again when <i>BT1</i> is pressed, the clock is running, and...
390	2 <sup>nd</sup>	...The <i>FOFO</i> buffer now...	...The <i>FIFO</i> buffer now...