Please note that Cypress is an Infineon Technologies Company.
The document following this cover page is marked as “Cypress” document as this is the company that originally developed the product. Please note that Infineon will continue to offer the product to new and existing customers as part of the Infineon product portfolio.

Continuity of document content
The fact that Infineon offers the following product as part of the Infineon product portfolio does not lead to any changes to this document. Future revisions will occur when appropriate, and any changes will be set out on the document history page.

Continuity of ordering part numbers
Infineon continues to support existing part numbers. Please continue to use the ordering part numbers listed in the datasheet for ordering.



THIS SPEC IS OBSOLETE

Spec No: 002-04935

Spec Title: AN204935 - EE PROM INTERFACE ROUTINES FOR XICOR X25080 USING THE SYNCHRONOUS SERIAL INTERFACE (SSI)

Replaced By: NONE

AN204935

EE PROM Interface Routines for Xicor X25080 Using the Synchronous Serial Interface (SSI)

This Application Note describes the EE PROM Interface Routines for Xicor X25080 using the Synchronous Serial Interface (SSI) and provides some useful subroutine functions to handle the data transfer.

OBSOLETE

Contents

1 Introduction....1
2 Talking to Xicor....2
2.1 Benefits....3
A Appendix A....4
B Appendix B....10
Document History....14

1 Introduction

The SSI interface is a rather simple peripheral function found on all 8-bit Microcontrollers.

It basically consists of a simple 8-bit shift register and a control register. (See Figure. 1 SSI Basic Block Diagram)

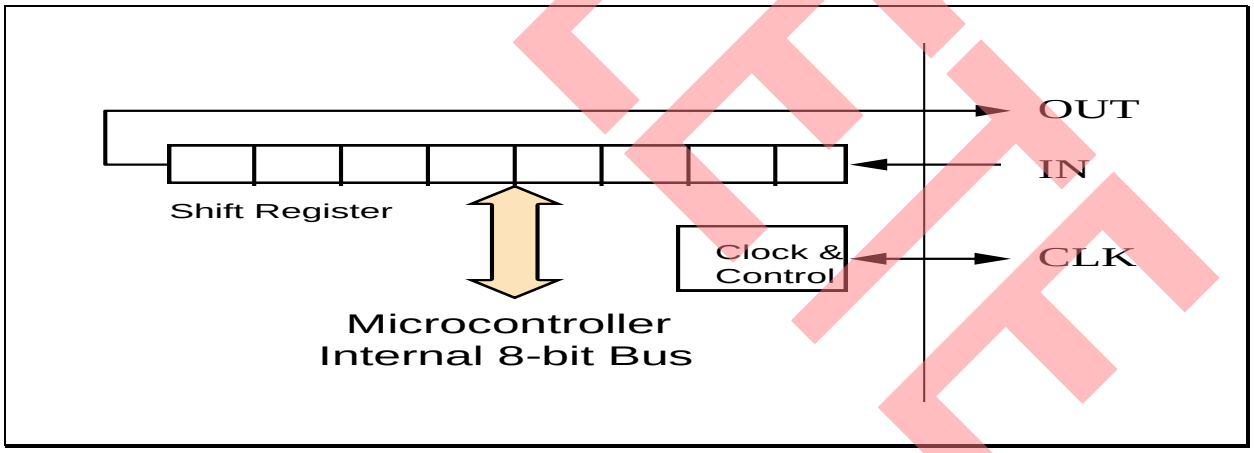
The controller software can byte-wise read from or write to the shift register.

When setting a start bit in the control register, the 8-bit data is shifted within 8 clock cycles.

At each clock cycle, one bit-information will appear at the output signal OUT.

At the same time, the level on the input signal IN will be shifted into the shift register at the other end.

Figure. 1 SSI Basic Block Diagram



The clock signal can be generated by the microcontroller (Master Mode), or can be provided externally (Slave Mode). The shift direction, MSB first or LSB first can also be configured.

Parameters which cannot be configured are the fixed data-length of always 8-bits, and the timing protocol.

The basic SSI timing diagram is given in Figure. 2 Basic Timing Diagram.

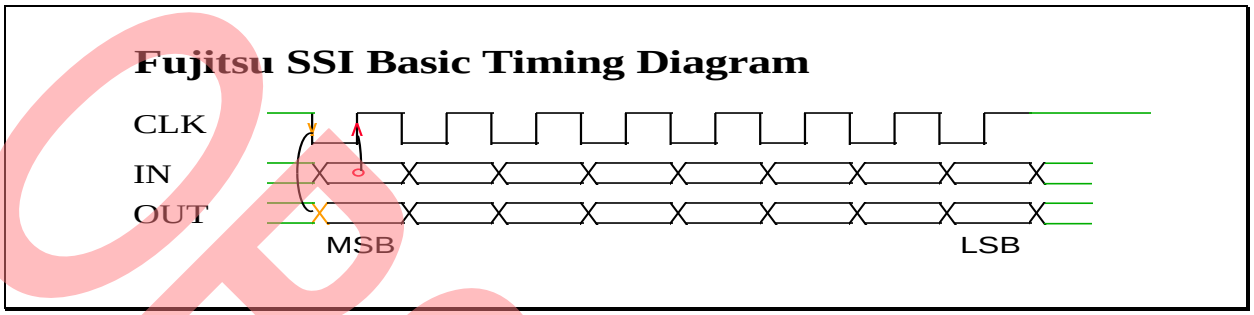
On the SSI interface, once a data transfer is started, the clock signal will go from its inactive Hi-level to Lo-level. The output signals OUT will be set according to the bit content at the falling edge.

At the following rising edge, (Lo to Hi transition) the SSI interface will sample the input signal IN.

This way, for example two microcontrollers could be directly connected and exchange data.

(Of course IN and OUT must be cross-connected and one must work in master and the other in slave mode.)

Figure. 2 Basic Timing Diagram



When it comes to connecting other things like serial EEPROMs, this timing does not always fit.

For example to connect a SPI™ (Serial Peripheral Interface) compatible NM25C04 from National, an external inverter would be needed to invert the clock signal output.

Of course, such problems can be worked around by implementing the serial interface only in software, using some standard I/O lines.

This would also be the only solution if the serial protocol is even more complicated like the MICROWIRE™ protocol from National, which is not always based on a multiple of 8-bits data format.

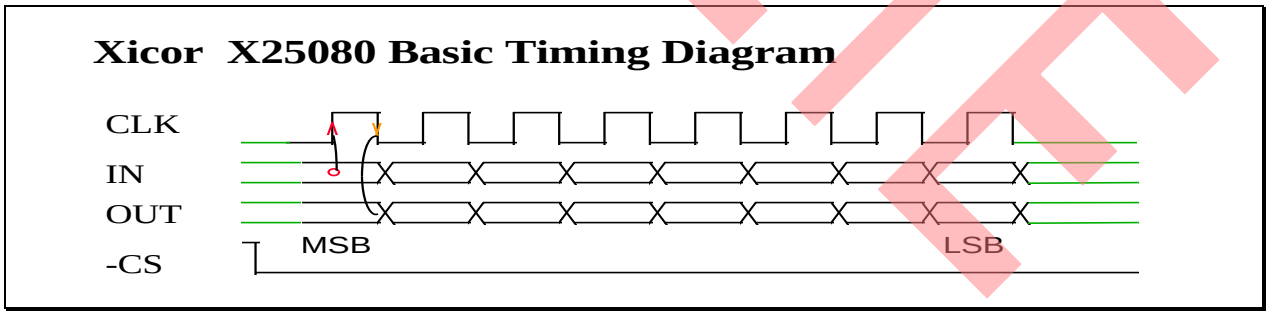
Obviously a parallel to serial conversion and vice versa implemented in software is significantly slower than utilizing a shift register.

2 Talking to Xicor

Fortunately some of the Xicor EEPROMs (which by the way also conform to one of the SPI modes) fits with the Cypress protocol, so they can be directly connected to the SSI interface without additional logic.

The Xicor protocol does not comply 100% with the SSI timing, as can be seen by comparing Figure. 2 Basic Timing Diagram and Figure. 3 X25080 Basic Timing Diagram, but technically this difference does not matter.

Figure. 3 X25080 Basic Timing Diagram



The only difference is, that the clock signal for the EEPROM should be at low level at the beginning and at the end of a transfer (the inactive time when chip select is at high level).

OBSOLETE

If directly connected to the Cypress SSI interface, this will not be the case since the SSI interface will keep the clock signal at a high level during inactive time.

In practice, this difference does not matter as tests have been proven.

Important is, that data outputs on the SSI and EEPROM outputs change at the falling clock edges,

and data inputs on the SSI and EEPROM inputs take place at rising clock edges, and this is still the case.

Typical data transfer to/from the EEPROM usually consist of quite long serial words, but fortunately

the number of bits is always a multiple of 8-bits, so the data transfer can be established by a number of single 8-bit transfers, using the SSI interface.

For example, to read some bytes out of the EEPROM, the controller would have to do the following steps:

1. Activate the EEPROM chip select signal by a standard output pin.
2. Send a “Read Command” byte to the EEPROM (1st Byte Transfer (send))
3. Send the read start address to the EEPROM (HiByte) (2nd Byte Transfer (send))
4. “ (LoByte) (3rd Byte Transfer (send))
5. Read the 1st data byte (4th Byte Transfer (receive))

x. “ following bytes if required (x receive))

y. Deactivate the EEPROM chip select signal again.

2.1 Benefits

Of course, the main advantage of using the SSI is, that it allows faster transfers compared to a pure software solution.

The following table gives some comparison between software and SSI supported interface functions, based on a maximum main clock speed of 10Mhz.

The maximum serial bit rate is a more theoretical figure, since also for the SSI supported version, some software overhead is necessary to transfer data to the serial shift register and to monitor the control register.

The other figures present more realistic figures measured from activating the EEPROM chip select until deactivating it again.

	Bytes Transf.	Software Implementation (Rotate, Bit Instructions, etc.)	SSI supported Implementation (Clock Speed = CLK/2)
Maximum Serial Bit Rate	-	9,5 µs	0,8 µs
Read Status Operation	2	180 µs	25 µs
Read One Data Byte	4	370 µs	60 µs
Read 10 Data Bytes	14	1200 µs	230 µs

The “EE.asm” example program in appandix A, demonstrates the SSI / EEPROM interface routines on a MB89630 Evaluation Kit.

The “EES.asm” contains some of these functions in a pure software implementation and was used to demonstrate the difference in performance.

OBSOLETE

A Appendix A

```
$lo cy xr cp

        NAME   "XICOR"                  ; module name
 &SET DebugFunctions 0                  ; Flag: Do Not Include Debug Functions
 &INCLUDE "c:\FJ_8L_3\INCLUDE\eBIOS.inc";       just BIOS Macros

SYCC    EQU  7                          ; System Speed Controll

;==     Some Port Definitions   ==
CHG3    EQU  00Dh                       ; Port 3 Change Register
PDR3    EQU  00Dh                       ; Port 3 Data Register
DDR3    EQU  00Eh                       ; Port 3 Direction Register
SMR1    EQU  01Ch                       ; Serial 1 Mode register
SDR1    EQU  01Dh                       ; Serial 1 Data register
ILR1    EQU  07Ch                       ; Interrupt Controler Level Register

;===    Protocoll Equates       ===
X_RdSr          EQU     H'05            ; Cmd : Read Status
X_WrEn          EQU     H'06            ;       Write Enable
X_Read          EQU     H'03            ;       Read Data
X_Write         EQU     H'02            ;       Write Data

X_IN            EQU     5               ; SCI-Interface Bit Definitions
X_OUT           EQU     4
X_CLK           EQU     3
X_CS            EQU     0

;----------------------------------------
;--     Dummy Segment Definitions       --
;----------------------------------------
DIRVAR  DIRSEG
        RB  1
DIRVAR  ENDS
;----------------------------------------
;----------------------------------------
DVAR    DSEG
Str     RB 32
DVAR    ENDS
```

OBSOLETE

```
;****************************************
;--     Demo Program Reset Entry        --
;****************************************
Code    CSEG ABS
        ORG H'2000

Reset:  MOV   SYCC,#H'1F                ; Hi Speed MainClock
        MOVW  SP,#0280H                 ; Init Stack Pointer (Internal RAM)
        MOVW  A,#H'0030                 ; IL = 3 !
        MOVW  PS,A                      ; Init PS register
        PRSTR DemoProgMsg               ; Print Initial Message

Main:
        CALL  InitSSI                   ; Init SSI-Interface
        CALL  StartMsg                  ;

;       CALL  BenchMark                 ; Read 1 and 10 Bytes (Benchmark)

        CALL  X_WriteEnable
        CALL  X_ReadStatus              ; Read Status Byte
        PrHexB SDR1

        ;== Interactive Demonstration of Read / Write Strings ==
MainL:  PRSTR MainLoop
        RdChar
        CMP   A,#'r'
        BNE   CkWr
        PRSTR ReadMsg                   ;-- REAR String from EEPROM --
        MOVW  EP,#Str                   ; Target Address
        MOVW  A,#0                      ; EE-PROM Source Address
        MOV   R0,#32                    ; max. 32 characters
        CALL  X_ReadData

        PRSTR Str
        JMP   MainL

CkWr:   CMP   A,#'w'
        BNE   MainL
        PRSTR WriteMsg                  ;-- WRITE String into EEPROM --
        RDSTR Str,32
        MOVW  EP,#Str
        CALL  StrLen                    ; compute length of String
        MOV   R0,A                      ; copy actual Str.Len into R0
        MOVW  A,#0                      ; EE-PROM Target Address
        MOVW  EP,#Str                   ; Controller Source Address
        CALL  X_WriteData
        JMP   MainL

Stop:   NOP
        JMP   Stop

StrLen: PUSHW IX
FindEnd:MOV   A,@EP
        BEQ   GotLen
        INCW  EP
        INCW  IX
        JMP   FindEnd
GotLen: MOVW  A,IX
        POPW  IX
        RET
```

OBSOLETE

```
StartMsg:
        PRSTR Started
        RdChar
        RET

;----------------------------------------
;--      Benchmark Tests                --
;----------------------------------------
BenchMark:
        CALL   ReadOneByte              ; Performance Test 1
        CALL   StartMsg                 ;
        CALL   ReadTenByte              ; Performance Test 2
        JMP    StartMsg                 ;
ReadOneByte:
        MOVW   A,#0                     ; EE-PROM Source Address
        MOVW   EP,#Str                  ; Target Address
        MOV    R0,#1                    ; max. 32 characters
        JMP    X_ReadData
ReadTenByte:
        MOVW   A,#0                     ; EE-PROM Source Address
        MOVW   EP,#Str                  ; Target Address
        MOV    R0,#10                   ; max. 32 characters
        JMP    X_ReadData
```

OBSOLETE

```
;=========================================
;==   Xicor X25080 Interface Routines   ==
;=========================================
        ;-------------------------------------------------
        ;--     Initialise Port(CS) and SSI-Interface   --
        ;-------------------------------------------------
InitSSI:
        MOV  PDR3,#H'FD                 ; DeActiveate CS
        MOV  DDR3,#H'03                 ; P30 = ChipSelect = Output
        MOV  SMR1,#B'00110010           ; SCK,-SO Output, Int.Clock, MSB 1st
        RET

        ;-------------------------------------------------
        ;--     Transfer Write Enable Command           --
        ;-------------------------------------------------
X_WriteEnable:
        CLRB PDR3:X_CS                  ; Activate CS-EEPROM
        MOV  SDR1,#X_WrEn               ; Write Enable Command
        SETB SMR1:0                     ; Start Transfer
WaTra0: BBC  SMR1:7,WaTra0              ; wait until Transfer completed
        CLRB SMR1:7                     ; clear interrupt flag
        SETB PDR3:X_CS                  ; DeActivate CS-EEPROM
        RET

        ;-------------------------------------------------
        ;--     Read Status Register                    --
        ;-------------------------------------------------
X_ReadStatus:
        CLRB PDR3:X_CS                  ; Activate CS-EEPROM
        MOV  SDR1,#X_RdSr               ; Read Status Command
        SETB SMR1:0                     ; Start Transfer
WaTra1: NOP
        BBC  SMR1:7,WaTra1              ; wait until Transfer completed
        CLRB SMR1:7                     ; clear interrupt flag
        SETB SMR1:0                     ; Start Transfer 2nd Time
WaTra2: NOP
        BBC  SMR1:7,WaTra2              ; wait until Transfer completed
        CLRB SMR1:7                     ; clear interrupt flag
        MOV  A,SDR1                     ; read input
        SETB PDR3:X_CS                  ; DeActivate CS-EEPROM
        RET

        ;-------------------------------------------------
        ;--     Read Bytes from EEPROM                  --
        ;--     EP   : Target Address                   --
        ;--     A    : EEPROM Source Address            --
        ;--     R0   : Byte Count                       --
        ;-------------------------------------------------
X_ReadData:
        CLRB PDR3:X_CS                  ; Activate CS-EEPROM
        MOV  SDR1,#X_Read               ; Read Data Command
        SETB SMR1:0                     ; Start Transfer
WaTra3: BBC  SMR1:7,WaTra3              ; wait until Transfer completed
        CLRB SMR1:7                     ; clear interrupt flag
        SWAP
        MOV  SDR1,A                     ; Transfer MS-Address Byte
        SETB SMR1:0                     ; Start Transfer
WaTra4: BBC  SMR1:7,WaTra4              ; wait until Transfer completed
        CLRB SMR1:7                     ; clear interrupt flag
        SWAP
```

OBSOLETE

```
        MOV  SDR1,A                     ; Transfer LS-Address Byte
        SETB SMR1:0                     ; Start Transfer
WaTra5: BBC  SMR1:7,WaTra5              ; wait until Transfer completed
        CLRB SMR1:7                     ; clear interrupt flag
RTraLop:MOV  SDR1,#0                    ; just dummy
        SETB SMR1:0                     ; Start Data Transfer
WaTra6: BBC  SMR1:7,WaTra6              ; wait until Transfer completed
        CLRB SMR1:7                     ; clear interrupt flag
        MOV  A,SDR1                     ; get received byte
        MOV  @EP,A                      ; store received byte
        INCW EP
        DEC  R0
        BNE  RTraLop                    ; continue
        SETB PDR3:X_CS                  ; DeActivate CS-EEPROM
        RET

        ;-------------------------------------------------
        ;--     Write Bytes to EEPROM (Max 32 Bytes)    --
        ;--     EP  : Controller Souce Address          --
        ;--     A   : EEPROM Target Address             --
        ;--     R0  : Byte Count                        --
        ;-------------------------------------------------
X_WriteData:
        CLRB PDR3:X_CS                  ; Activate CS-EEPROM
        MOV  SDR1,#X_Write              ; Write Data Command
        SETB SMR1:0                     ; Start Transfer
WaTra7: BBC  SMR1:7,WaTra7              ; wait until Transfer completed
        CLRB SMR1:7                     ; clear interrupt flag
        SWAP
        MOV  SDR1,A                     ; Transfer MS-Address Byte
        SETB SMR1:0                     ; Start Transfer
WaTra8: BBC  SMR1:7,WaTra8              ; wait until Transfer completed
        CLRB SMR1:7                     ; clear interrupt flag
        SWAP
        MOV  SDR1,A                     ; Transfer LS-Address Byte
        SETB SMR1:0                     ; Start Transfer
WaTra9: BBC  SMR1:7,WaTra9              ; wait until Transfer completed
        CLRB SMR1:7                     ; clear interrupt flag
WTraLop:MOV  A,@EP                      ; load Byte
        INCW EP
        MOV  SDR1,A                     ;
        SETB SMR1:0                     ; Start Data Transfer
WaTraA: BBC  SMR1:7,WaTraA              ; wait until Transfer completed
        CLRB SMR1:7                     ; clear interrupt flag
        DEC  R0
        BNE  WTraLop                    ; continue
        SETB PDR3:X_CS                  ; DeActivate CS-EEPROM
        RET

;-----------------------------------------
;--     Some Const. Data                --
;-----------------------------------------
DemoProgMsg     DB 13,10
                DB 13,10,"*****************************"
                DB 13,10,"** XiCor EE-PROM Functions **"
                DB 13,10,"*****************************",13,10,0
Started         DB 13,10,"PDR3-Interface Initialized"
                DB 13,10,"(Start Logic Analizer for Timing Measurement)",0
MainLoop        DB 13,10
                DB 13,10,"Command Loop ('R' : ReadEEStr, 'W' : WriteEEStr)",0
```

OBSOLETE

```
ReadMsg         DB 13,10,"R: ",0
WriteMsg        DB 13,10,"W:>",0
Done            DB " End ",0

;----------------------------------------
;--     Some Symbol Info for Monitor    --
;----------------------------------------
SymTab  DB "SymTab:",0          ; Symbol Table Header
        DB LabL1-LabS1,0        ; Length of 1st Symbol, dummy byte
LabS1   DB "Reset"              ; Symbol Name
LabL1   DW Reset                ; Symbol Value
        DB LabL2-LabS2,0
LabS2   DB "Main"
LabL2   DW Main
        DB LabL3-LabS3,0
LabS3   DB "Stop"
LabL3   DW Stop
        DB 00                   ; End of Symbol Table
        DB 01                   ; 0 BreakPoint
        DW Stop
Code    ENDS

;----------------------------------------
;--     Symbol Table Entry              --
;----------------------------------------
SVector cseg ABS
        ORG BIOST+24h           ; Pointer to Symbol Table
        DW  SymTab
SVector ends

;----------------------------------------
;--     Reset Vector            --
;----------------------------------------
RVector CSEG ABS                ;
        ORG H'FFFC              ;
        DB  0                   ;
        DB  1                   ;
ResVec  DW  Reset               ;
RVector ends                    ;
        END
```

OBSOLETE

B Appendix B

```
$lo cy xr cp

        NAME    "XICOR"                 ; module name
 &SET DebugFunctions 0                  ; Flag: Do Not Include Debug Functions
 &INCLUDE "c:\FJ_8L_3\INCLUDE\eBIOS.inc";       just BIOS Macros

SYCC    EQU  7                          ; System Speed Controll

;==     Some Port Definitions   ==
CHG3    EQU  00Dh                       ; Port 3 Change Register
PDR3    EQU  00Dh                       ; Port 3 Data Register
DDR3    EQU  00Eh                       ; Port 3 Direction Register
SMR1    EQU  01Ch                       ; Serial 1 Mode register
SDR1    EQU  01Dh                       ; Serial 1 Data register
ILR1    EQU  07Ch                       ; Interrupt Controler Level Register

;===    Protocoll Equates       ===
X_RdSr          EQU     H'05            ; Cmd : Read Status
X_WrEn          EQU     H'06            ;       Write Enable
X_Read          EQU     H'03            ;       Read Data
X_Write         EQU     H'02            ;       Write Data

X_IN            EQU     5               ; SCI-Interface Bit Definitions
X_OUT           EQU     4
X_CLK           EQU     3
X_CS            EQU     0

;----------------------------------------
;--     Dummy Segment Definitions       --
;----------------------------------------
DIRVAR  DIRSEG
        RB  1
DIRVAR  ENDS
;----------------------------------------
;----------------------------------------
DVAR    DSEG
Str     RB 32
DVAR    ENDS

;****************************************
;--     Demo Program Reset Entry        --
;****************************************
Code    CSEG ABS
        ORG H'2000

Reset:  MOV   SYCC,#H'1F                ; Hi Speed MainClock
        MOVW  SP,#0280H                 ; Init Stack Pointer (Internal RAM)
        MOVW  A,#H'0030                 ; IL = 3 !
        MOVW  PS,A                      ; Init PS register
        PRSTR DemoProgMsg               ; Print Initial Message
Main:
        CALL  InitSSI                   ; Init SSI-Interface
        CALL  StartMsg                  ;

;       CALL  BenchMark                 ; Read 1 and 10 Bytes (Benchmark)

        CALL  X_ReadStatus              ; Read EE-Prom Status
        MOV   R0,A
```

OBSOLETE

```
        PrHexB R0

        PRSTR PrMsg
        MOVW  A,#0                      ; EE-PROM Source Address
        MOVW  EP,#Str                   ; Target Address
        MOV   R0,#32                    ; max. 32 characters
        CALL  X_ReadData
        PRSTR Str

Stop:   NOP
        JMP   Stop
StartMsg:
        PRSTR Started
        RdChar
        RET

;---------------------------------------
;--     Benchmark Tests                --
;---------------------------------------
BenchMark:
        CALL  ReadOneByte               ; Performance Test 1
        CALL  StartMsg                  ;
        CALL  ReadTenByte               ; Performance Test 2
        JMP   StartMsg                  ;
ReadOneByte:
        MOVW  A,#H'0000                 ; EE-PROM Source Address
        MOVW  EP,#Str                   ; Target Address
        MOV   R0,#1                     ; max. 32 characters
        JMP   X_ReadData
ReadTenByte:
        MOVW  A,#0                      ; EE-PROM Source Address
        MOVW  EP,#Str                   ; Target Address
        MOV   R0,#10                    ; max. 32 characters
        JMP   X_ReadData


;=======================================
;==   Xicor X25080 Interface Routines   ==
;=======================================
        ;-----------------------------------------------
        ;--     Initialise Port(CS,CLK)                --
        ;-----------------------------------------------
InitSSI:
        MOV  PDR3,#H'F3                 ; DeActiveate CS, CLK=Lo
        MOV  DDR3,#H'1F                 ; P30..P34 = Output
        RET
        ;-----------------------------------------------
        ;-- Software Parallel to Serial (Tx) Conversion --
        ;-----------------------------------------------
TxByte: MOV  R1,#8
TxLop:  ROLC A
        BC   SetBit
ClrBit: CLRB PDR3:X_OUT
        BNC  NxTxB
SetBit: SETB PDR3:X_OUT
NxTxB:  SETB PDR3:X_CLK                 ; Generate Positive Edge
        CLRB PDR3:X_CLK                 ; Generate Negative Edge
        DEC  R1
        BNE  TxLop
        RET
```

OBSOLETE

```
        ;-----------------------------------------------
        ;-- Software Serial to Parallel (Rx) Conversion --
        ;-----------------------------------------------
RxByte: MOV  R1,#8
RxLop:  BBS  PDR3:X_IN,BitSet
BitClr: CLRC
        BNC  RxBil
BitSet: SETC
RxBil:  ROLC A
        SETB PDR3:X_CLK                 ; Generate Positive Edge
        CLRB PDR3:X_CLK                 ; Generate Negative Edge
        DEC  R1
        BNE  RxLop
        RET
        ;-----------------------------------------------
        ;--      Read Status Register                 --
        ;-----------------------------------------------
X_ReadStatus:
        CLRB PDR3:X_CS                  ; Activate CS-EEPROM
        MOV  A,#X_RdSr                  ; Read Status Command
        CALL TxByte                     ; Tx Byte
        CALL RxByte
        SETB PDR3:X_CS                  ; DeActivate CS-EEPROM
        RET
        ;-----------------------------------------------
        ;--      Read Bytes from EEPROM               --
        ;--      EP  : Target Address                 --
        ;--      A   : EEPROM Source Address          --
        ;--      R0  : Byte Count                     --
        ;-----------------------------------------------
X_ReadData:
        CLRB PDR3:X_CS                  ; Activate CS-EEPROM
        MOV  A,#X_Read                  ; Read Data Command     (LoAdr in TL)
        CALL TxByte                     ; Tx Byte
        SWAP                            ; HiAdr into AL
        CALL TxByte                     ; Tx Byte
        XCH  A,T                        ; restore LoAdr
        CALL TxByte                     ; Tx Byte
RTraLop:CALL RxByte
        MOV  @EP,A                      ; store received byte
        INCW EP
        DEC  R0
        BNE  RTraLop                    ; continue
        SETB PDR3:X_CS                  ; DeActivate CS-EEPROM
        RET

;----------------------------------------
;--      Some Const. Data              --
;----------------------------------------
DemoProgMsg     DB 13,10
                DB 13,10,"****************************"
                DB 13,10,"** XiCor EE-PROM Functions **"
                DB 13,10,"****************************",13,10,0

Started         DB 13,10,"PDR3-Interface Initialized"
                DB 13,10,"(Start Logic Analizer for Timing Measurement)",0
PrMsg           DB 13,10,"EEPROM : ",0
;----------------------------------------
;--      Some Symbol Info for Monitor  --
;----------------------------------------
```

OBSOLETE

```
SymTab  DB "SymTab:",0          ; Symbol Table Header
        DB LabL1-LabS1,0        ; Length of 1st Symbol, dummy byte
LabS1   DB "Reset"              ; Symbol Name
LabL1   DW Reset                ; Symbol Value
        DB LabL2-LabS2,0
LabS2   DB "Main"
LabL2   DW Main
        DB LabL3-LabS3,0
LabS3   DB "Stop"
LabL3   DW Stop
        DB 00                   ; End of Symbol Table
        DB 00                   ; 0 BreakPoint
Code    ENDS
;---------------------------------------
;--     Symbol Table Entry              --
;---------------------------------------
SVector cseg ABS
        ORG BIOST+24h           ; Pointer to Symbol Table
        DW  SymTab
SVector ends
;---------------------------------------
;--     Reset Vector            --
;---------------------------------------
RVector CSEG ABS                ;
        ORG H'FFFC              ;
        DB  0                   ;
        DB  1                   ;
ResVec  DW  Reset               ;
RVector ends                    ;
        END
```

OBSOLETE

Document History

Document Title: AN204935 - EE PROM Interface Routines for Xicor X25080 Using the Synchronous Serial Interface (SSI)

Document Number: 002-04935

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	—	WOFR	02/20/2014	Initial release.
*A	5333834	WOFR	07/01/2016	Migrated Fujitsu Application Note from MCU-AN-389013-E-V10 to Cypress format.
*B	5612311	WOFR	01/31/2017	Spec obsoleted, no further updates planned.

OBSOLETE

Worldwide Sales and Design Support

Cypress maintains a worldwide network of offices, solution centers, manufacturer's representatives, and distributors. To find the office closest to you, visit us at Cypress Locations.

Products

ARM® Cortex® Microcontrollers	cypress.com/arm
Automotive	cypress.com/automotive
Clocks & Buffers	cypress.com/clocks
Interface	cypress.com/interface
Lighting & Power Control	cypress.com/powerpsoc
Memory	cypress.com/memory
PSoC	cypress.com/psoc
Touch Sensing	cypress.com/touch
USB Controllers	cypress.com/usb
Wireless/RF	cypress.com/wireless

PSoC® Solutions

PSoC 1 | PSoC 3 | PSoC 4 | PSoC 5LP

Cypress Developer Community

Forums | Projects | Videos | Blogs | Training | Components

Technical Support

cypress.com/support

OBSOLETE

PSoC is a registered trademark and PSoC Creator is a trademark of Cypress Semiconductor Corporation. All other trademarks or registered trademarks referenced herein are the property of their respective owners.

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709

Phone : 408-943-2600
Fax : 408-943-4730
Website : www.cypress.com

© Cypress Semiconductor Corporation,2014-2017. This document is the property of Cypress Semiconductor Corporation and its subsidiaries, including Spansion LLC ("Cypress"). This document, including any software or firmware included or referenced in this document ("Software"), is owned by Cypress under the intellectual property laws and treaties of the United States and other countries worldwide. Cypress reserves all rights under such laws and treaties and does not, except as specifically stated in this paragraph, grant any license under its patents, copyrights, trademarks, or other intellectual property rights. If the Software is not accompanied by a license agreement and you do not otherwise have a written agreement with Cypress governing the use of the Software, then Cypress hereby grants you a personal, non-exclusive, nontransferable license (without the right to sublicense) (1) under its copyright rights in the Software (a) for Software provided in source code form, to modify and reproduce the Software solely for use with Cypress hardware products, only internally within your organization, and (b) to distribute the Software in binary code form externally to end users (either directly or indirectly through resellers and distributors), solely for use on Cypress hardware product units, and (2) under those claims of Cypress's patents that are infringed by the Software (as provided by Cypress, unmodified) to make, use, distribute, and import the Software solely for use with Cypress hardware products. Any other use, reproduction, modification, translation, or compilation of the Software is prohibited.

TO THE EXTENT PERMITTED BY APPLICABLE LAW, CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS DOCUMENT OR ANY SOFTWARE OR ACCOMPANYING HARDWARE, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. To the extent permitted by applicable law, Cypress reserves the right to make changes to this document without further notice. Cypress does not assume any liability arising out of the application or use of any product or circuit described in this document. Any information provided in this document, including any sample design information or programming code, is provided only for reference purposes. It is the responsibility of the user of this document to properly design, program, and test the functionality and safety of any application made of this information and any resulting product. Cypress products are not designed, intended, or authorized for use as critical components in systems designed or intended for the operation of weapons, weapons systems, nuclear installations, life-support devices or systems, other medical devices or systems (including resuscitation equipment and surgical implants), pollution control or hazardous substances management, or other uses where the failure of the device or system could cause personal injury, death, or property damage ("Unintended Uses"). A critical component is any component of a device or system whose failure to perform can be reasonably expected to cause the failure of the device or system, or to affect its safety or effectiveness. Cypress is not liable, in whole or in part, and you shall and hereby do release Cypress from any claim, damage, or other liability arising from or related to all Unintended Uses of Cypress products. You shall indemnify and hold Cypress harmless from and against all claims, costs, damages, and other liabilities, including claims for personal injury or death, arising from or related to any Unintended Uses of Cypress products.

Cypress, the Cypress logo, Spansion, the Spansion logo, and combinations thereof, PSoC, CapSense, EZ-USB, F-RAM, and Traveo are trademarks or registered trademarks of Cypress in the United States and other countries. For a more complete list of Cypress trademarks, visit cypress.com. Other names and brands may be claimed as property of their respective owners.