AN1222
APPLICATION NOTE

I^2C Communication between ST52x420 and EEPROM

Authors: V. Marino, C. Vinci

1. Introduction

This application note shows an example of how to use ST52x420 to communicate with an EEPROM memory with an I^2C bus protocol. In this example, an M24C04 EEPROM (4K bit) is taken into account, but the following considerations can be applied to applications with any kind of M24CXX memory.

Two software routines are proposed, the Byte Write and the Random Address Read, for ST52x420 microcontroller configured in single master communication.

2. Communicating Protocol

The M24CXX is an EEPROM supporting the I^2C protocol. The M24CXX can communicate with a microcontroller, the master, only by a serial data I/O line (SDA) and a serial clock (SCL).

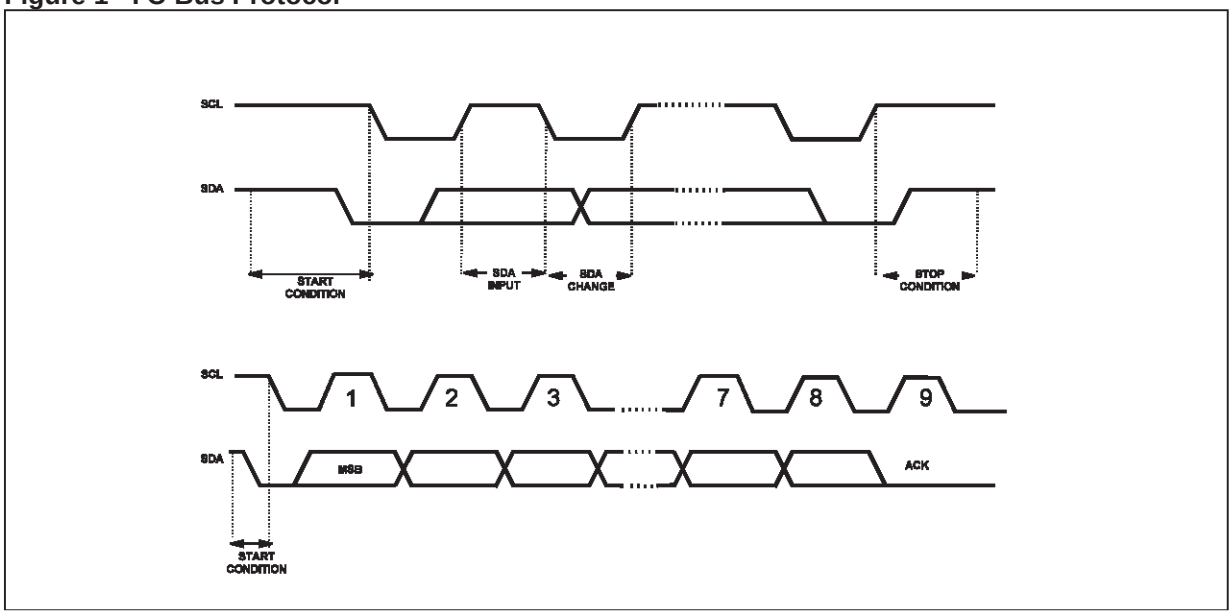
During each data transfer, the M24CXX samples the SDA bus signal on the rising edge of the clock signal SCL. The SDA signal must be stable during the clock low to high transition and the data must change only when the SCL line is low. Changes in the data line while the clock is high are interpreted as START and STOP conditions.

START is identified by a high to low transition of the SDA line while the clock is stable in the high state. A START condition must precede any data transfer command.

After the START, ST52x420 sends onto the SDA bus line 8 bits (MSB first): the first 7 bits to select the device, the last (*RW* bit) to indicate if it is a read (*RW* high) or write (*RW* low) operation.

After sending each 8 bits data stream, the master releases the SDA bus; during the 9th clock pulse period the receiver pulls the SDA bus low to acknowledge the receipt of the 8 bit data. A complete data transfer is always terminated by a STOP, identified by a low to high transition of the SDA line while the clock SCL is stable in the high state.

Figure 1 - I^2C Bus Protocol



3. Hardware Description

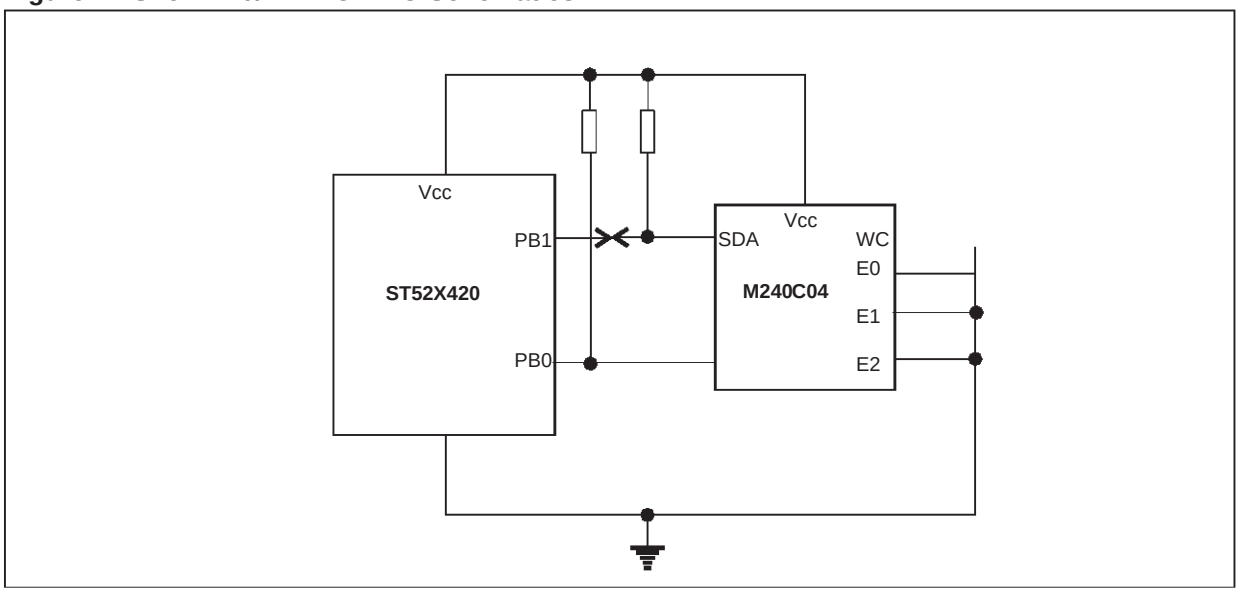
The connection scheme between ST52x420 and EEPROM is shown in Figure 2. Pin PB1 of ST52x420 is used to transfer data to and from the memory (SDA); pin PB0 for data synchronization (SCL).

In this scheme M24C04 inputs E0, E1 and E2 are tied to Vss (Device Select Code is A0h), therefore, being E0 low, only 256 byte of memory, the low part, can be addressed. To address also the memory high part (address 1xxxxxxxx), E0 must be dynamically driven by using another I/O pin of ST52x420.

A maximum of four memories of the type M24C04 can be addressed by a microcontroller on the same two wire buses, setting E1 and E2 inputs (00, 01, 10, 11): each device is identified by its Device Select Code and will only respond to the correct selection.

Pin $\overline{WC}$, used to protect the contents of the memory from inadvertent erase/write operations, is unconnected, therefore with this scheme, write operations are always allowed. To allow enabling (WC=V_{IL}) and disabling WC=V_{IH}) write operations, WC must be driven dynamically.

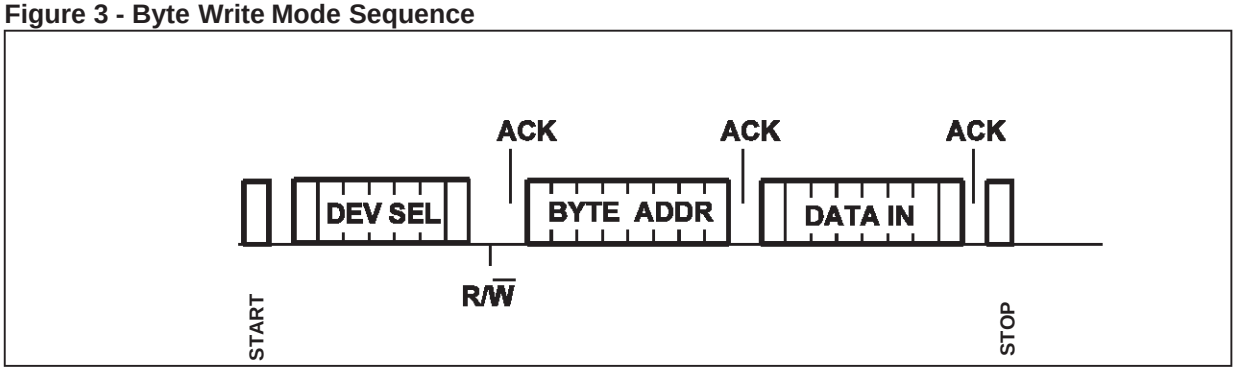
Figure 2 - ST52x420/EEPROM I²C Schematics



3.1 Byte Write Mode

In WRITE mode, after the START condition, ST52x420 sends a Device Select Code with the $R\overline{W}$ bit set to '0', as shown in Figure 3. The memory acknowledges the reception and waits for an address byte from the master, to which it responds with an acknowledge. After the acknowledge, ST52x420 sends the data byte to be written in the defined memory location. ST52x420 terminates the transfer by generating a STOP condition.

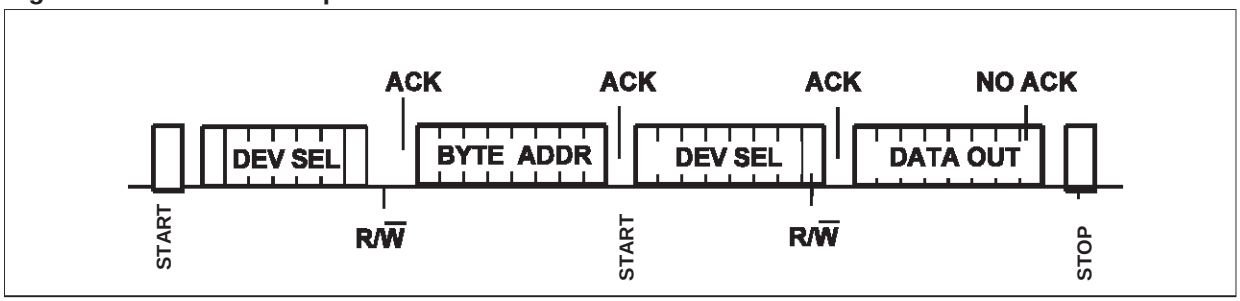
Figure 3 - Byte Write Mode Sequence



3.2 Random Address Read Mode

In order to read a byte from an address of the memory, ST52x420 performs a dummy write to load the address, as shown in Figure 4. Then, without sending a STOP condition, ST52x420 sends another START condition and sends the Device Select Code again, with the *RW* bit set to '1'. After acknowledgment, the memory provides onto the bus the contents of the addressed byte. The master is not required to acknowledge the byte output and terminates the transfer with a STOP condition.

Figure 4 - Read Mode Sequence



4. Software (I²C communication routines between ST52x420 and M24C04 EEPROM)

The software project for ST52x420 communicating with an EEPROM M24C04 is described in the followings. Two routines are developed in assembler language to show how to perform operations of writing and reading a page of memory (256 bytes). Please refer to Appendix 1 for writing operation and Appendix 2 for reading operation assembler code.

4.1 Write Software Routine

Chip configuration and EPROM data store

In the first part of the assembler code, as you can see in Appendix 1, after the interrupt request vector definition, the "data page addr value" instructions are performed. These instructions indicate to store data value in the ST52x420 EPROM, in the address of the specified page, in order to write these data into I²C EEPROM.

The configuration registers used for this program are:

REG_CONF0 to enable TIMER1 interrupt

REG_CONF1 to establish interrupt priority

REG_CONF8 and REG_CONF9 to configure the TIMER1

REG_CONF13 and REG_CONF14 to configure the PORTB

The Timer1 peripheral has been configured in order to have a timer clock T=0.4µs by setting the prescaler value to 4 and supposing that the master clock frequency is equal to 10 Mhz. The choice to load the counting from the timer_count register, where the user can write values ranging from 10 to 255 (80 in this example), allows to set the communication speed within the range [250KHz (fast mode), 9.8KHz (standard mode)].

Main program

After the initialization of the used register, the START condition is performed calling the subroutine '*start_bit*'.

Then three byte are serially sent (calling '*send_data*') onto SDA bus, respectively the Device Select Code, the Address byte and the Data byte to be written into I²C memory address location: after each byte is sent, ST52X420 waits for M24C04 acknowledgment; at each ACK receipt (reg30 is 0 after '*read_ack*' subroutine) the variable '*cont_ack*' (reg40), initially set to zero, is incremented of one unit. When '*cont_ack*' reaches 3, a STOP condition (calling '*stop_bit'* subroutine) is sent by ST52X420 to the memory. If the signal ACK is not returned from the memory (reg30 not equal to zero) the program provides to restart the communication protocol, jumping to the label '*init'*.

In order to write 256 bytes into the memory, the reg10 (address counter) is incremented of one unit after each 'byte write' operation and only when the whole 256 bytes have moved to the EEPROM the program terminates with a '**halt**' instruction, that stops the clock master so that the CPU and the peripherals are turned-off.

'Start_bit' subroutine

This subroutine performs the START condition: at first, the SDA line is pulled in high state through sending the reg21=2=00000010b into port B; then the SCL line is pulled high by setting pin PB1 high sending reg21=3=00000011b**;** finally SDA is put low, and then also SCL, to complete the START bit communication, as required from the protocol (Figure 1).

Time delays among transitions between each block in the '*start_bit*' routine are managed thanks to the timer1 peripheral. In fact each transition on the port B is performed after a '***waiti***' instruction, that allows to synchronize the signal into port B with the interrupt generated on the falling edge of the Timer1 out signal.

'Send_data' subroutine

This routine has been planned for sending serially the 8 bits of a byte.

In the first part of the routine, to decide whether the Divice Select Code byte (160 in this case), or the address byte or the data byte must be sent, the variable '*cont_ack*' is used.

If the byte to be sent is the data byte, an operation of reading from the EPROM of ST52X420 is performed, through the instruction '**ldre (19) (10)'**, that loads in the register of RAM pointed by reg19 (that is reg20) the content of the address of memory pointed by reg10 (that is 256*page+reg10).

To send each byte, stored in reg50, bit by bit, a cycle scanned by reg23 (decremented from 8 to 1) is performed. At each step of the cycle, one data bit is sent onto SDA bus ('**ldpr 1 21**') and the clock pulse on SCL line is performed.

To determine if the bit to send is '0' or '1', a simple operation of shift on the reg50 ('**asl 50**') is done.

'Read_ack' subroutine

The routine '*read_ack*' is called by the main program after the routine '*send_data*' to check if the data sent is correctly received by the EEPROM; if so, the variable '*ack*' (reg30) is 'zero' and the procedure can continue, otherwise, if reg30 is 'two', an error has occurred and the communication is restarted.

'Stop_bit' subroutine

A data transfer is always terminated by a STOP condition, that is identified by a low to high transition of SDA line while the clock SCL is stable in the high state. This condition is performed calling '*stop_bit*' subroutine.

4.2 Read Software Routine

Main program

After the initialization of the used register, the START condition is performed calling the subroutine '*start_bit*'.

Three bytes are then sent serially onto SDA bus (see also '*send_data*' subroutine). A cycle variable '*cont_ack*' (reg40) is used to discriminate which byte is going to be sent: the Device Select Code (160) if reg40 is '0', the address byte (reg10) if it is '1', the Device Select Code again with the R*/W* bit set to '1' (i.e. A byte corresponding to 161), after a new START condition, if reg40 is '2'. If reg40 is '3', the data is read (with '*read_data*' subroutine) and the STOP condition is executed.

Note that, if the signal ACK is not returned from the memory (reg30 not equal to zero) the program provides to restart the communication protocol, jumping to the label '*init*'.

In order to read from the memory 256 bytes the reg10 (address counter) is incremented of one unit after each read operation and only when the whole 256 bytes have read from the EEPROM the program terminates with a '**halt**' instruction, that stops the clock master so that the CPU and the peripherals are turned-off.

'Read_data' subroutine

This subroutine is designated to read the data from the memory; the data is composed of eight bits, then, in order to read it, a cycle scanned by a counter initially set to '8' (reg23) is realized.

To convert the serial data to parallel format, is necessary to associate to every bit the relative weight, that is given by $2^{(reg23-1)}$; to calculate the weight, the cycle '*power_of_two*' is performed. At the end of this cycle, the register 31 of the RAM will contain 0 if the read bit was '0' or a value among 128, 64, 32, 16, 8, 4, 2, 1 if the bit was '1'; adding to reg20 reg31 at every cycle, the value of the read data is finally contained in reg20.

'Send_ack' subroutine

In accordance with the I^2C communicating protocol, at the end of the data byte receipt, ST52X420 puts the pin PB0 (SCL line) high, before sending the stop condition.

Appendix 1

```
;*************************************************************
;
;   PURPOSE:        I2C communication
;
;    DATE:          12/1/2000
;
;*************************************************************
; This program writes from the address 0 to the address 255 of
; an I2C the bytes contained to the page 2 of the EPROM of ST52X420

;*************************Interrupt vector*********************
irq 2 TIM1
;*************************************************************

;data to store in the EPROM

data    2       0       255
data    2       1       254
data    2       2       253
data    2       3       252
data    2       4       251
data    2       5       250
…       …       …       …
…       …       …       …
data    2       249     6
data    2       250     5
data    2       251     4
data    2       252     3
data    2       253     2
data    2       254     1
data    2       255     0
;*************************************************************
;*****************Peripherals and Chip configurations *********
        ldrc    0       00001000b       ; interrupt mask configuration
        ldcr    0       0               ; TM1 not masked, TM0, TM2, ADC, EXT
                                          masked
        ldrc    0       00011110b       ;  interrupt priority configuration
        ldcr    1       0               ; from the lowest: ADC, TM0, TM2, TM1

        ldrc    0       01000000b       ; PWM-Timer 1 configuration
        ldcr    8       0               ; Timer mode, int. on TM1Out falling edge
        ldrc    0       00100010b       ; PWM-Timer 1 configuration
```

```
        ldcr    9       0               ; TM1Out square wave, PRESC1=4
        ldrc    0       00000010b       ; Port B direction configuration
        ldcr    13      0               ; PB1 input, the whole other pins in  output
        ldrc    0       00000000b       ;Port B mode configuration
        ldcr    14      0               ; PB7-PB0 digital I/O

        ldrc    0       80              ; reg0=80
        ldpr    5       0               ; TIMER_COUNT1=80

;*************************************endconfiguration**************************

;*************************************main program*****************************
;*************************************variablesdefinition**********************
;TM1 manages the communication speed
;PB0SCL   PB1SDA
;reg10=address
;reg20=data
;reg30=ack
;reg40=cont_ack
;reg50=byte_to_send
;*************************************************************************
Start:
        ldrc    10      0               ;address=0
        ldrc    19      20              ;reg19=20 (this instruction maids for  the indirect
                                        ;addressing of reg20 through reg19)
        ldrc    21      0               ;register of support to send data into  Port B
        ldrc    22      2               ; register used to set bit1 of Port B

init:
        ldrc    40      0               ; init cont_ack
        call start_bit                  ; START condition routine

send:
        call send_data                  ; send byte into SDA bus
        call read_ack                   ; read ACK (result is stored in reg30)

        and     30      30              ; if ack=0 the procedure goes on, else restarts
        jpnz  init
        inc     40                      ; increments cont_ack

        ldrc    41      3               ; after three acknowledge the data transfer is
        sub     41      40              ; terminated and a STOP condition is performed
        jpnz send                       ; call stop_bit
                                        ; STOP condition routine
        inc     10                      ; increments address
                                        ; If before this operation reg10<255 flag C is
```

```
                                        ; cleared and then new write operation is  performed.
                                        ; Else, if reg10=255, this operation causes an
                                        ; overflow (flag C is setted) and the program

        jpnc    init                    ; finishes.
        halt                            ; clock master is stopped
;*****************************end main program*************************

;*****************************start_bitroutine*************************

;this routine performs the start of the communicating protocol

start_bit:
        ldrc    0       00000000b       ; Port B direction configuration
        ldcr    13      0               ; PB6-PB0 output

        ldrc    0       01000101b       ; PWM-Timer 1 configuration
        ldcr    8       0               ; Timer set and start

        waiti                           ; waits for Timer1 interrupt

;*****************************SDAhigh**********************************
        add     21      22              ; reg21=2
        ldpr    1       21              ; send reg21 into Port B
;**********************************************************************

        waiti                           ; waits for Timer1 interrupt

;*****************************SCLhigh**********************************
        inc     21                      ; reg21=3
        ldpr    1       21              ; send reg21 into Port B
;**********************************************************************
        waiti                           ; waits for Timer1 interrupt

;*****************************SDAlow***********************************
        sub     21      22              ; reg21=1
        ldpr    1       21              ; send reg21 into Port B
;**********************************************************************

        waiti                           ; waits for Timer1 interrupt

;*****************************SCLlow***********************************
        dec     21                      ; reg21=0
        ldpr    1       21              ; send reg21 into Port B
;**********************************************************************
```

```
        waiti                                   ; waits for Timer1 interrupt

        ldrc    0       01000000b       ; PWM-Timer 1 configuration
        ldcr    8       0               ; Timer reset

        ldrc    0       00000010b       ; Port B direction configuration
        ldcr    13      0               ; PB1 input, the whole other pins in output
        ret                             ; return from subroutine

;*****************************************************************************

;*****************************send_dataroutine*********************

send_data:

        ldrc    0       00000000b       ; Port B direction configuration
        ldcr    13      0               ; PB6-PB0 output

;*****************************byte_to_send*************************

        ldrc    0       0               ; if cont_ack==0 then
        sub     0       40
        jpnz    l0

        ldrc    50      160             ; byte_to_send=160 (Device Select Code)
        jp      end     0

l0:                                     ; else
        ldrc    0       1               ; if cont_ack==1 then
        sub     0       40
        jpnz    l1

        ldrr    50      10              ; byte_to_send=address
        jp      end0

l1:                                     ; else

        pgset   2                       ; sets the current EPROM page

        ldre (19) (10)                  ; this instruction loads into RAM location, whose
                                        ; address is contained in reg19 (that is into reg20)
                        ; the contents of the EPROM location to the
                                        ; address (256*pag_number+reg10)
        ldrr    50      20              ; byte_to_send=data
end0:
```

```
;**********************************************************************
;**********************************************************************

        ldrc    23      8               ; init counter bit

        bit:

;*************************bitto send*****************************
        ldrc    21      0               ; bit=0
        asl     50                      ; shift left byte_to_send
        jpnc    l2                      ; C is set if MSB was high
        add     21      22              ; if C is set bit=1 (PB1=1)
l2:
        ldpr    1       21              ; send bit of data into Port B

;**********************************************************************

;*********************SCLpulse*****************************
        ldrc    0       01000101b       ; PWM-Timer 1 configuration
        ldcr    8       0               ; Timer set and start

        waiti                           ; waits for Timer1 interrupt

        inc     21
        ldpr    1       21              ; SCL high

        waiti                           ; waits for Timer1 interrupt
        dec     21
        ldpr    1       21              ; SCL low

        waiti                           ; waits for Timer1 interrupt

        ldrc    0       01000000b       ; PWM-Timer 1 configuration
        ldcr    8       0               ; Timer reset
;**********************************************************************

        ldrc    21      0               ; resets reg21

        dec     23                      ; decrements counter bit
        jpnz  bit                       ; jumps to label 'bit' if the data transfer is
                                        ; incomplete, goes out after 8 bits sent
;*************************************************
        ldrc    0       00000010b       ; Port B direction configuration
        ldcr    13      0               ; PB1 input, the whole other pins in output
```

```
        ret                                     ; return from subroutine

;*************************************************

;***********read_ack routine**********************

read_ack:

        ldrc    0       01000101b       ; PWM-Timer 1 configuration
        ldcr    8       0               ; Timer set and start

        waiti                           ; waits for Timer1 interrupt

;*************************************SCL high ***********************************
        inc     21                      ; reg21=1
        ldpr    1       21              ; send reg21 into Port B
;*************************************************************************

waiti                                   ; waits for Timer1 interrupt

        ldri    30      10              ; read port B
        and     30      22              ; (XXXXXXBX)b AND (00000010)b=(000000B0)b

;*************************************SCL low ************************************
dec     21                              ; reg21=0
        ldpr    1       21              ; send reg21 into Port B
;*************************************************************************

        waiti                           ; waits for Timer1 interrupt

        ldrc    0       01000000b       ; PWM-Timer 1 configuration
        ldcr    8       0               ; Timer reset
        ldrc    21      0               ; resets reg21

ret                                     ; return from subroutine
;*************************************************************************
;*************************stop_bit routine********************************

stop_bit:

        ldrc    0       00000000b       ; Port B direction configuration
        ldcr    13      0               ; PB6-PB0 output

        ldrc    0       01000101b       ; PWM-Timer 1 configuration
        ldcr    8       0               ; Timer set and start
```

```
            waiti                                   ; waits for Timer1 interrupt
;***********SDA low*************************************************************************
            ldrc        21          0               ; reg21=0
            ldpr        1           21              ; send reg21 into Port B
;***************************************************************************************

            waiti                                   ; waits for Timer1 interrupt

;***********SCL high************************************************************************
            inc         21                          ; reg21=1
            ldpr        1           21              ; send reg21 into Port B
;***************************************************************************************
            waiti                                   ; waits for Timer1 interrupt
;***********SDA high************************************************************************
            add         21          22              ; reg21=3
            ldpr        1           21              ; send reg21 into Port B
;***************************************************************************************

            waiti                                   ; waits for Timer1 interrupt
;***********SCL low*************************************************************************
            dec         21                          ; reg21=2
            ldpr        1           2               ; send reg21 into Port B
;***************************************************************************************

            waiti                                   ; waits for Timer1 interrupt

            ldrc        0           01000000b       ; PWM-Timer 1 configuration
            ldcr        8           0               ; Timer reset

            ldrc        21          0               ; resets reg21

            ldrc        0           00000010b       ; Port B direction configuration
            ldcr        13          0               ; PB1 input, the whole other pins in output
;*****************************INTs Subroutines**************************************************
TIM1:
            reti                                    ; return from interrupt
;***************************************************************************************
```

Appendix 2

```
;*************************************************************************
;
;   PURPOSE:        I2C communication
;
;   DATE:           12/1/2000
;
;*************************************************************************
;this program reads from an I2C the byte from the address 0 to the address 255
;*************************interrupt vectors ***************************
irq 2 TIM1
;*************************************************************************

;******** Peripherals and Chip configurations ***************************

        ldrc    0       00001000b       ; interrupt mask configuration
        ldcr    0       0               ; TM1 not masked, TM2, TM0, ADC, EXT masked

        ldrc    0       00011110b       ; interrupt priority configuration
        ldcr    1       0               ; from the lowest: ADC,TM0,TM2,TM1

        ldrc    0       00000000b       ; Port A configuration
        ldcr    4       0               ; PA7-PA0 output

        ldrc    0       01000000b       ; PWM-Timer 1 configuration
        ldcr    8       0               ; Timer mode, int. on TM1Out falling edge


        ldrc    0       00100010b       ; PWM-Timer 1 configuration
        ldcr    9       0               ; TM1Out square wave, PRESC1=4

        ldrc    0       00001000b       ; Port A mode configuration
        ldcr    12      0               ; Port A 8 bits, pin22-PA3, pin23-PA2, pin24-PA1

        ldrc    0       00000010b       ; Port B direction configuration
        ldcr    13      0               ; PB1 input, the whole other pins in output

        ldrc    0       00000000b       ; Port B mode configuration
        ldcr    14      0               ; PB7-PB0 digital I/O
        ldrc    0       80              ; reg0=80
        ldpr    5       0               ; TIMER_COUNT1=80

;*************************************************************************
```

```
;*************************************mainprogram*************************************
;*************************************Variablesdefinition*************************************
;TM1 manages the communication speed
;PB0SCL   PB1SDA
;reg10=address
;reg20=data
;reg30=ack
;reg40=cont_ack
;reg50=byte_to_send
;*************************************************************************************

start:
        ldrc    10      0       ; address=0
        ldrc    21      0       ; register of support to send data into Port B
        ldrc    22      2       ; register used to set bit1 of Port B


init:
        ldrc    40      0       ; init cont_ack

start:
        call start_bit          ; START condition routine

send:
        call send_data          ; send byte into SDA bus
        call read_ack           ; read ACK (result is stored in reg30)

        and     30      30      ; if ack=0 the procedure goes on, else restarts
        jpnz init

        inc     40              ; increments cont_ack
        ldrc    41      1       ; if cont_ack=1 then jumps to label 'send' in order
        sub     41      40      ; to send another byte into SDA bus
        jpz send

        ldrc    41      2       ; if cont_ack=2 then jumps to label 'start'in order
        sub     41      40      ; to perform another START condition
        jpz start

        call read_data          ; read procedure
        call send_ack           ; send acknowledgment
        call stop_bit           ; STOP condition routine
```

```
        ldpr    0       20      ; send read data into port A

        inc     10              ; increments address
                                ; If before this operation reg10<255 flag C is
                                ; cleared and then new read operation is performed.
                                ; Else, if reg10=255, this operation causes an
                                ; overflow (flag C is set) and the program
        jpnc    init            ; finishes.

        halt                    ; clock master is stopped

;**********end main program**********

;**********start_bit routine**********

;this routine performs the start of the communicating protocol

start_bit:

        ldrc    0       00000000b       ; Port B direction configuration
        ldcr    13      0               ; PB6-PB0 output

        ldrc    0       01000101b       ; PWM-Timer 1 configuration
        ldcr    8       0               ; Timer set and start

        waiti                           ; waits for Timer1 interrupt

;**********SDA high**********
        add     21      22              ; reg21=2
        ldpr    1       21              ; send reg21 into Port B
;**********

        waiti                           ; waits for Timer1 interrupt

;**********SCL high**********
        inc     21                      ; reg21=3
        ldpr    1       21              ; send reg21 into Port B
;**********

        waiti                           ; waits for Timer1 interrupt

;**********SDA low**********
        sub     21      22              ; reg21=1
        ldpr    1       21              ; send reg21 into Port B
```

```
;*************************************************************************************************

        waiti                                   ; waits for Timer1 interrupt

;*************************************************SCL low  ***************************************
        dec     21              ; reg21=0
        ldpr    1       21                      ; send reg21 into Port B
;*************************************************************************************************

        waiti                                   ; waits for Timer1 interrupt

        ldrc    0       01000000b               ; PWM-Timer 1 configuration
        ldcr    8       0                       ; Timer reset

        ldrc    0       00000010b               ; Port B direction configuration
        ldcr    13      0                       ; PB1 input, the whole other pins in output

        ret                                     ; return from subroutine

;*************************************************************************************************

;*************************************************send_dataroutine*********************************

send_data:

        ldrc    0       00000000b               ; Port B direction configuration
        ldcr    13      0                       ; PB6-PB0 output

;*************************************************byte_to_send ************************************

        ldrc    0       0                       ; if cont_ack==0 then
        sub     0       40
        jpnz    l0

        ldrc    50      160                     ; byte_to_send=160 (Device Select Code, RW=0)
        jp end0
l0:                                             ; else
        ldrc    0       1                       ; if cont_ack==1 then
        sub     0       40
        jpnz    l1

        ldrr    50      10                      ; byte_to_send=address
```

```
        jp      end0

l1:                                     ; else
        ldrc    50      161             ; byte_to_send=161 (Device Select Code, RW=1)
end0:
;*************************************************************************************

;*************************************************************************************
        ldrc    23      8               ; init counter bit

bit:

;*************************************************************************************
        ldrc    21      0               ; bit=0
        asl     50                      ; shift left byte_to_send
        jpnc    l2                      ; C is set if MSB was high
        add     21       22             ; if C is set bit=1 (PB1=1)
l2:
        ldpr    1       21              ;send bit of data into PORT B

;*************************************************************************************


;*************************************SCL pulse ***************************************
        ldrc    0       01000101b       ; PWM-Timer 1 configuration
        ldcr    8       0               ; Timer set and start

        waiti                           ; waits for Timer1 interrupt

        inc     21
        ldpr    1       21              ; SCL high

        waiti                           ; waits for Timer1 interrupt
        dec     21
        ldpr    1       21              ; SCL low

        waiti                           ; waits for Timer1 interrupt

        ldrc    0       01000000b       ; PWM-Timer 1 configuration
        ldcr    8       0               ; Timer reset
;*************************************************************************************

        ldrc    21      0               ; resets reg21

        dec     23                      ; decrements counter bit
```

```
        jpnz  bit                       ; jumps to label 'bit' if the data transfer is
                                        ; incomplete, goes out after 8 bits sent
;*****************************************************************************

        ldrc    0       00000010b       ; Port B direction configuration
        ldcr    13      0               ; PB1 input, the whole other pins in output

        ret                             ; return from subroutine
;*****************************************************************************


;*****************************read_ackroutine*********************************

read_ack:

        ldrc    0       01000101b       ; PWM-Timer 1 configuration
        ldcr    8       0               ; Timer set and start

        waiti                           ; waits for Timer1 interrupt

;*****************************SCLhigh *****************************************
        inc     21                      ; reg21=1
        ldpr    1       21              ; send reg21 into Port B
;*****************************************************************************

        waiti                           ; waits for Timer1 interrupt

        ldri    30      10              ; read port B
        and     30      22              ; (XXXXXXBX)b AND (00000010)b=(000000B0)b
;*****************************SCLlow  *****************************************
        dec     21                      ; reg21=0
        ldpr    1       21              ; send reg21 into Port B
;*****************************************************************************

        Waiti                           ; waits for Timer1 interrupt

        ldrc    0       01000000b       ; PWM-Timer 1 configuration
        ldcr    8       0               ; Timer reset
        ldrc    21      0               ; resets reg21
        ret                             ; return from subroutine
;*****************************************************************************

;*****************************read_dataroutine*********************************

read_data:
```

```
        ldrc    20      0               ; reg20=0 (reg20 will contain the read data)
        ldrc    23      8               ; init counter bit of data to read

;*********************************************************************************************
read_bit:
        ldrc    0       01000101b       ; PWM-Timer 1 configuration
        ldcr    8       0               ; Timer set and start

        waiti                           ; waits for Timer1 interrupt
        inc     21                      ; reg21=1
        ldpr    1 21                    ; SCL high

        waiti                           ; waits for Timer1 interrupt

        ldri    31      10              ; read port B
        and     31      22              ; (XXXXXXBX)b AND (00000010)b=(000000B0)b
        asr     31                      ; reg31=(0000000B)b
        ldrr    32      23              ; reg32=counter bit
;*********************************************************************************************

;*********************************************************************************************

power_of_two:

        dec     32                      ; this loop shifts left reg31 so many times
        jpz end 1                       ; how much determines the weight of the read bit
        asl     31                      ; (2^(reg32-1))
        jp power_of_two

end1:
        add     20      31              ; reg20=b7*2^7 + b6*2^6 +...+ b0*2^0
;*********************************************************************************************

        dec      21                     ; reg21=0
        ldpr    1       21              ; SCL low

        waiti                           ; waits for Timer1 interrupt

        ldrc    0       01000000b       ; PWM-Timer 1 configuration
        ldcr    8       0               ; Timer reset

        dec     23                      ; decrement counter bit
        jpnz read_bit                   ; jump to label 'read_bit' if the data transfer is
                                        ; incomplete, goes out after 8 bits read
ldrc    21      0                       ; resets reg21
```

```
        ret                             ; return from subroutine

;************************************************************************************************

;*****************************************************send_ack routine*********************************************

end_ack:

        ldrc    0       00000000b       ; Port B direction configuration
        ldcr    13      0               ; PB6-PB0 output

        ldrc    0       01000101b       ; PWM-Timer 1 configuration
        ldcr    8       0               ; Timer set and start


        waiti                           ; waits for Timer1 interrupt

;*****************************************************SDA high**********************************************
        add     21      22              ; reg21=2
        ldpr    1       21              ; send reg21 into Port B
;************************************************************************************************
        waiti                           ; waits for Timer1 interrupt

;****************************************************SCL high ***********************************************
        inc     21                      ; reg21=3
        ldpr    1       21              ; send reg21 into Port B
;************************************************************************************************

        waiti                           ; waits for Timer1 interrupt

;****************************************************SCL low ************************************************
        dec     21                      ; reg21=2
        ldpr    1       21              ; send reg21 into Port B
;************************************************************************************************

        waiti                           ; waits for Timer1 interrupt

        ldrc    0       01000000b       ; PWM-Timer 1 configuration
        ldcr    8       0               ; Timer reset

        ldrc    21      0               ; resets reg21

        ldrc    0       00000010b       ; Port B direction configuration
```

```
        ldcr    13      0               ; PB1 input, the whole other pins in output

        ret                             ; return from subroutine

;***************************************************************************

;**********stop_bitroutine*****************************************

stop_bit:

        ldrc    0       00000000b       ; Port B direction configuration
        ldcr    13      0               ; PB6-PB0 output


        ldrc    0       01000101b       ; PWM-Timer 1 configuration
        ldcr    8       0               ; Timer set and start

        waiti                           ; waits for Timer1 interrupt

;**********************************SDA low ***************************************
        ldrc    21      0               ; reg21=0
        ldpr    1       21              ; send reg21 into Port B
;***************************************************************************

        waiti                           ; waits for Timer1 interrupt


;**********************************SCL high ***************************************
        inc     21                      ; reg21=1
        ldpr  1 21                      ; send reg21 into Port B
;***************************************************************************

        waiti                           ; waits for Timer1 interrupt

;**********************************SDA high ***************************************
        add     21      22              ; reg21=3
        ldpr    1       21              ; send reg21 into Port B
;***************************************************************************

        waiti                           ; waits for Timer1 interrupt

;**********************************SCL low  ***************************************
        dec     21                      ; reg21=2
        ldpr    1       21              ; send reg21 into Port B
```

```
;*************************************************************************************************

        waiti                                   ; waits for Timer1 interrupt


        ldrc    0       01000000b       ; PWM-Timer 1 configuration
        ldcr    8       0               ; Timer reset

        ldrc    21      0               ; resets reg21

        ldrc    0       00000010b       ; Port B direction configuration
        ldcr    13      0               ; PB1 input, the whole other pins in output


        ret                             ; return from subroutine

;*************************************************************************************************


;*************************************INTs Subroutines ***********************************************
        TIM1:
        reti                            ; return from interrupt
;*************************************************************************************************
```