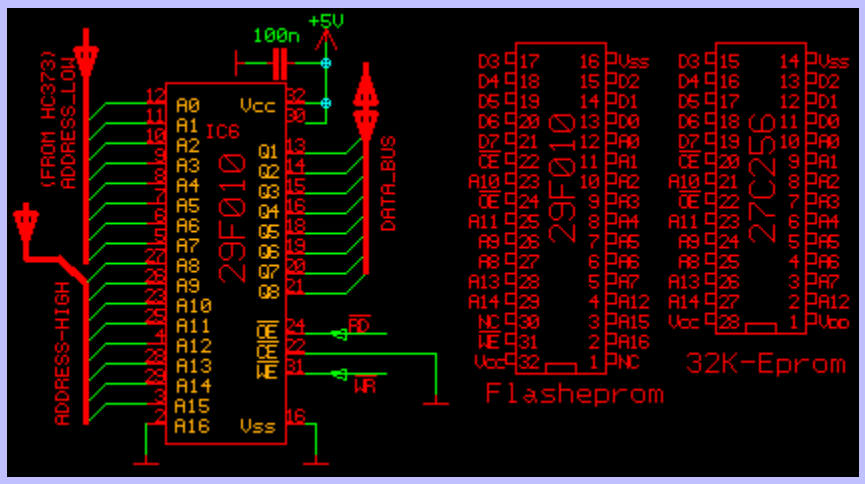
Orgler electronic

"Applications with microcontroller 68HC11"

FLASH AMD 29F010 with 68HC11



load-routine in assembler programs the FLASH 29F010
The board must be in BOOTSTRAP-MODE
WRFLASH.EXE This program request a BIN-FILE to program it
in the flash AMD29F010 with 68HC11 in BOOTSTRAP-MODE

First the program below is loaded in the RAM of 68HC11 and then this program switches the 68HC11 to Expanded-Multiplexed Mode and programs the bytes from BIN-FILE to FLASH.
The last instruction is a indirect JUMP
from RESET-VECTOR.
The 68HC11 starts the program in FLASH.

WRFLASH1.ZIP This is the 'C' source code for the program WRFLASH.EXE

```
*********************************************************
*       68HC11 programs the flash AMD29F010
*       1. HC11 expect char 'W'
*       2. HC11 expect startaddress (only 1 Byte -> HighByte)
*                                   lowbyte per default=0
*       3. action: clear FlashEprom
*       4. HC11 send $55
*          HC11 expect now the BINARY-FILE
*       5. HC11 send $AA for success
*       6. HC11 starts the program from RESET-VECTOR
*
*    Attention:
*      The program-code must be in the RAM of 68HC11
*********************************************************

FlashCommAdr1   EQU     $5555
FlashCommAdr2   EQU     $2AAA
```

```
CommReg1        EQU     $AA
CommReg2        EQU     $55

ResRead         EQU     $F0
AutoSel         EQU     $90
BytePro         EQU     $A0

EraseFlash      EQU     $80
EraseChip       EQU     $10

EraseSect       EQU     $30

hprio           EQU     $3C

*********************************************************
*include the 68HC11 ports addresses from $1000 to $103F
$HC11DEF.H

         ORG $0                                 code in RAM
START           LDS     #$FF
                LDX     #$1000
                BSET    hprio,X,#00100000B      SWITCH the 68HC11 in the
                BCLR    hprio,X,#00010000B      EXPANDED MULTIPLEXED MODE
                BCLR    hprio,X,#01000000B

                LDAA    #00110000B              9600 BAUD
                STAA    BAUD
                clra
                staa    SCCR1
                staa    SPCR
                staa    PORTD
                LDAA    #00001100B      68HC11 SCI enabled no interrupts
                STAA    SCCR2           RECEIVER & TRANS. ENABLED
                ldaa    #00110010B
                staa    DDRD
                ldaa    SCDAT

FirstByte       ldaa    SCSR
                anda    #00100000B
                beq     FirstByte

                ldaa    SCDAT
                cmpa    #'W'
                bne     FirstByte

WRITE_FLASH     jsr     RxBytetoA       * StartAddress HighByte
                clrb                    * StartAddress LowByte = 0;
                xgdx                    * X FlashAddress

                ldaa    $FFFE
                jsr     TxBytefromA
                ldaa    $FFFF
                jsr     TxBytefromA

                jsr     ERASE_FLASH       * erase Flash now

                ldaa    $FFFE
                jsr     TxBytefromA
                ldaa    $FFFF
                jsr     TxBytefromA
*---------------------------------------
FlashNext       bsr     RxBytetoA
```

```
               cpx     #$4000
               blo     NoProgFlash
               cpx     #$B800          internal EEPROM
               bhs     ProgFlash       from 68HC11A1
               cpx     #$B600
               bhs     NoProgFlash
*---------------------------------------
ProgFlash      bsr     FlashProgByte
*---------------------------------------
NoProgFlash    bsr     TxBytefromA
               inx
               bne     FlashNext

               ldaa    #$AA            * success
               bsr     TxBytefromA

               ldx     $FFFE           load the RESET-VECTOR of 68HC11
               jmp     X               and jump to the START of program
************************************************
RESET_READ     bsr     flashOn
               ldab    #$F0
               stab    FlashCommAdr1
               ldaa    $FFFE
               rts
*********************************************
ERASE_FLASH    ldab    #CommReg1
               stab    FlashCommAdr1
               ldab    #CommReg2
               stab    FlashCommAdr2
               ldab    #$80
               stab    FlashCommAdr1

               ldab    #CommReg1
               stab    FlashCommAdr1
               ldab    #CommReg2
               stab    FlashCommAdr2
               ldab    #$10
               stab    FlashCommAdr1
*              ---------------------Data Poll
D__PollWait    ldab    $FFFE
               bpl     D__PollWait
               rts
*********************************************
RxBytetoA      ldaa    SCSR
               anda    #00100000B
               beq     RxBytetoA
               ldaa    SCDAT
               rts
*              ---------------
TxBytefromA
SEND_BYTE      ldab    SCSR
               andb    #01000000B
               beq     SEND_BYTE
               staa    SCDAT
               rts
*****************************************************
DLY10          LDY     #$0B29       *10 msec delay
               bra     DLY_C
DLY1           LDY     #$011D        *1 msec delay
DLY_C          DEY                      4cycles
               BNE     DLY_C            3cycles  -> 3,5 uSec
               RTS
```

```
*****************************************************
*
*  Entry:   X -> address  A -> data
*
*  Write the data in A t address in X
*  DataPoll
*               ****************************************
FlashProgByte   bsr     flashOn
                ldab    #BytePro
                stab    FlashCommAdr1
*              -------------------- now Flash is write enabled
                staa    X
*               -------------------- Data Poll
www             tab
                eorb    X
                andb    #$80
                bne     www
                rts
*               ************************************
flashOn         ldab    #CommReg1
                stab    FlashCommAdr1
                ldab    #CommReg2
                stab    FlashCommAdr2
                rts
*****************************************************
```