How to Use the NM93C86A Serial EEPROM as a PC/ Laptop Detachable Printer File Memory Card (DPFMC)

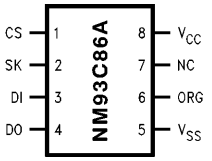
National Semiconductor
Application Note 936
Charles Watts
April 1994

INTRODUCTION

This applications note describes how to build a DPFMC. The card will be designed around a COP888CG microcontroller and four NM93C86A serial EEPROMs. The card will be designed to plug into any standard IBM-PC/Laptop parallel port. Once the card has been installed, the user can download any document (text, graphic, or combination) and print out that document at a later time. The DPFMC will be designed to make the computer think a printer is actually connected by simulating the printer's input port. Once all documents have been sent, the user needs to press the SEND-DOC button once to save the pointer address. Next the user can remove the card and turn its power off. The documents contained in the card's EEPROMs can be stored for hours, days, months, or even years if needed. However, hours will probably be a more realistic time frame. When the user is ready to print-out the documents saved, the card can be plugged into a stand along printer by either using the printer's DB-25 cable or (with an appropriate adapter) the printer itself. After switching on the cards power, all the user has to do is press the SEND DOC button and the printer will begin to print out the saved documents.

NM93C86A DESCRIPTION

The NM93C86A is a 16,384-bit CMOS non-volatile serial EEPROM that can be configured to have a 1024 x 16 or a 2048 x 8 architecture. The configuration is determined by the state of the ORG pin. If the ORG pin is tied low the NM93C86A is configured as a 2048-byte-wide memory. If the ORG pin is left floating or tied to V_{CC}, the 1024 word wide configuration is enabled. An internal pull-up resistor to V_{CC} assures that a floating ORG pin is pulled high. *Figure 1* shows the NM93C86A pin arrangement.



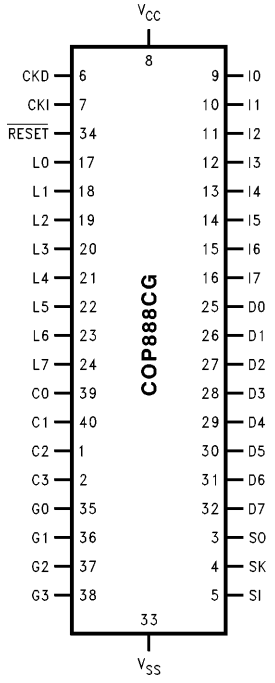
TL/D/12033–1

FIGURE 1. NM93C86A Pin Out

The NM93C86A has 7 instructions that can be performed. The instructions are: Read a byte/word (READ), Enable programming (EWEN), Disable programming (EWDS), Erase a byte/word (ERASE), Write a byte/word (WRITE), Erase all bytes/words (ERAL), and write a data pattern to all bytes/words (WALL). The NM93C86A uses the industry standard MICROWIRE™ interface.

COP888CG DESCRIPTION

The COP888CG is an 8-bit microcontroller. Its a fully static CMOS device containing RAM, ROM, and Microwire interface. The microcontroller contains 4,096 bytes of ROM used to store program code and 192 bytes of RAM used to store register data. It contains an 8-bit input, an 8-bit output, and two 8-bit bi-directional ports. The microcontroller also has a Microwire interface which will be used to connect the NM93C86A to it. These attributes make the COP888CG a good choice for this particular application.



TL/D/12033–2

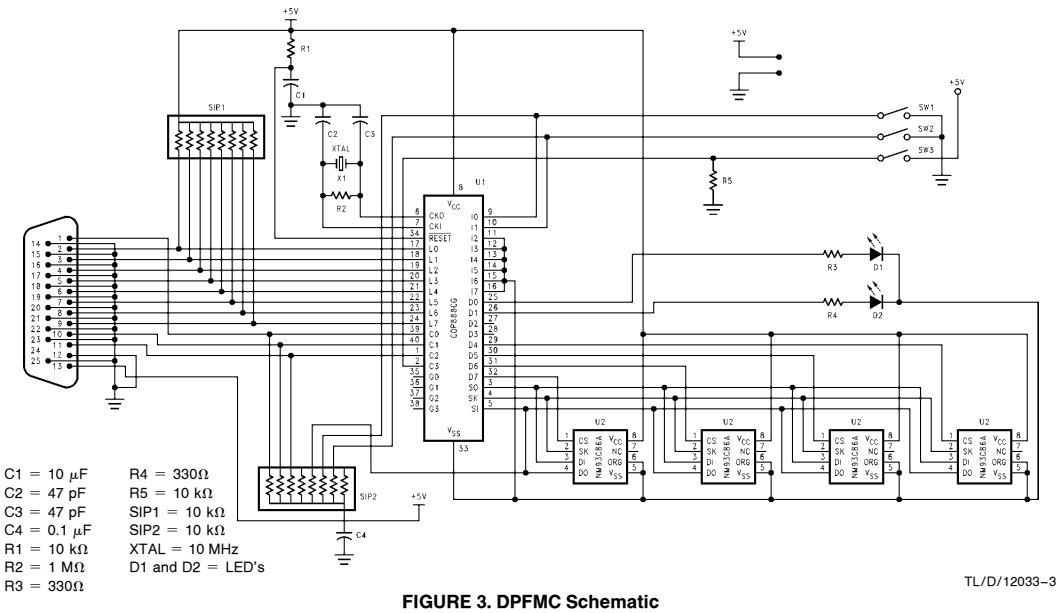
FIGURE 2. COP888CG Microcontroller

CIRCUIT DESCRIPTION

Figure 3 shows the complete schematic diagram of the DPFMC. The L-port of the microcontroller is connected to the D0–D7 data lines of the computer's bidirectional data port. The LS-nibble of C-port is connected to the STROBE, ACKNLG, BUSY, and the PE pins of the computer's parallel port. Pin 13 is tied HIGH and all of the other pins are tied LOW. All of the bi-directional I/O lines are pulled HIGH through two 10 kΩ SIP resistors.

The MICROWIRE Interface of the COP888CG uses pins 3(SO), 4(SK), and 5(SI). These pins are connect to the equivalent pins of the EEPROM. The MS-nibble of the D-port is used as chip select outputs to the four EEPROMs. Since the data the DPFMC will be processing is 1-byte length data, the NM93C86A will be configured to deal with 8-bit chunks instead of 16-bit chunks of data.

MICROWIRE™ is a trademark of National Semiconductor Corporation.



TL/D/12033–3

FIGURE 3. DPFMC Schematic

The DPFMC will have three inputs. Two are tied HIGH through 10 kΩ resistors. The third is tied LOW. The first input when LOW (pin 9) will indicate RECEIVE MODE. A LOW on the second input (pin 10) will indicate WRITE MODE. A HIGH on the third input (pin 11) will tell the microcontroller to save the current pointer address and begin sending the document data to the printer.

Their are also two LED outputs. LED1 will inform the user that the DPFMC cannot accept any more data. LED1 also informs the user that the pointer address has been saved. If LED1 flashes continuously the DPFMC is out of memory. If LED1 flashes irregular the DPFMC has saved the pointer address and is ready for power to be removed. LED2 informs the user of the current mode. Two flashes indicates the READ MODE and one flash indicates the WRITE MODE.

SOFTWARE DESCRIPTION

For simplicity and structure the software part of this application will be divided into three parts. The first or main block will monitor the three inputs and decide which mode the DPFMC will enter. The second block will control the interface logic between the computer's parallel port and the DPFMC.

In the case of a read operation, this block of code will start by configuring the STROBE and BUSY pins as inputs and the ACKNLG pin as an output. The routine then waits for the BUSY pin to fall LOW. When the BUSY pin falls low, the routine will begin monitoring the STROBE pin for a 0.5 µs LOW. When this happens, the routine reads the data port and stores that data into accumulator A. After the data is safely stored into accumulator A, the ACKNLG pin will be pulsed LOW for 5 µs to inform the computer that the data was received. Control is now passed to the final routine. This routine takes the data from ACC A stores the data into the EEPROMs. This routine basically will control the memory matrix part of the card. After the data is stored into the EEPROMs, control is passed back to the interface routine and the loop continues.

The write sequence will be just the opposite. The BUSY and STROBE pins are first configured to be outputs and the ACKNLG pin configured to be a input. The routine will then wait for the SEND DOC input to go low. When this happens the BUSY will go LOW to indicate the card is about to send a byte to the printer. The memory matrix routine the then stores the first byte of data into accumulator A. After that the interface routine sends that data to the port. The STROBE pin is pulsed LOW for 5 µs. From this point on the routine monitors the ACKNLG pin for a 5 µs LOW. When a LOW has occurred the routine loops back to the top, fetches the next byte, and starts a write loop.

```
; ASSEMBLY CODE FOe THE DETACHABLE PRINTER FILE MEMORY CARD(DPFMC)
; By Charles Watts
.INCLD COP888CG.INC
.SECT CODE,ROM,ABS=0
; ------------------- INITIALIZE PORT & REGISTER DATA ---------
DLYL   = 0F0
DLYH   = 0F1
ADDL   = 0F2
ADDH   = 0F3
BYT    = 00
HLD    = 01
STOLO  = 02
STOHI  = 03
STOHLD = 04
;
START: LD    PORTD, #00
       LD    PORTGC, #030
       SBIT  MSEL, CNTRL
       SBIT  S0,     CNTRL
       LD    PORTLC, #00
       LD    PORTCC, #0B
       LD    PORTCD, #02
       LD    B, #PORTCP
       JSR   LAST
; ------------------------- MAIN ROUTINE ------------------
MAIN:  LD    A, PORTI      ;
       IFEQ  A, #06        ;
       JSR   READ          ;
       IFEQ  A, #05        ;
       JSR   WRITE         ;
       JP    MAIN          ;
; ------------------------ SUBROUTINES WILL FOLLOW --------
READ:  LD    PORTLC, #00   ;CONFIGURE PORT
       LD    PORTCC, #06   ;TO READ MODE
       LD    PORTCD, #02   ;
       JSR   FLSH2         ;BLINK LED 2 TIMES
       JSR   EWEN          ;
LP1:   IFBIT 3, [B]        ;
       JSR   SAVE          ;
       IFBIT 0, [B]        ;WAIT FOR STROBE TO GO LOW
       JP    LP1           ;
       LD    PORTCD, #06   ;BRING BUSY HIGH
       LD    A, PORTLP     ;READ PORT AND SAVE IN ACCA
       X     A, BYT        ;
       JSR   PUT           ;STORE ACCA IN NVM
       LD    PORTCD, #04   ;PULSE ACKNLG
       NOP                 ;
       NOP                 ;
       LD    PORTCD, #06   ;
       NOP                 ;
       NOP                 ;
       LD    PORTCD, #02   ;
       JP    LP1           ;
;
WRITE: LD    PORTLC, #0FF  ;CONFIGURE PORT
       LD    PORTCC, #01   ;TO WRITE MODE
       LD    PORTCD, #01   ;
       JSR   FLSH1         ;
LP2:   IFBIT 3, [B]        ;
       JP    LP3           ;
```

TL/D/12033–4

```
        JP    LP2           ;
LP3:    IFBIT 2, [B]        ;WAIT FOR BUSY LOW
        JP    LP3           ;
        JSR   GET           ;GET BYTE FROM
        LD    A, BYT        ;
        X     A, PORTLD     ;NVM.
        NOP                 ;
        NOP                 ;
        LD    PORTCD, #00   ;PULSE STROBE
        NOP                 ;
        NOP                 ;
        LD    PORTCD, #01   ;
        NOP                 ;
        NOP                 ;
        JP    LP3           ;
;
GET:    LD    A, HLD        ;SET CS HIGH
        X     A, PORTD      ;
        LD    A, ADDH       ;SEND OPCOCE AND
        OR    A, #030       ;HI ADDRESS
        X     A, SIOR       ;
        SBIT  BUSY, PSW     ;
LP4:    IFBIT BUSY, PSW     ;
        JP    LP4           ;
        LD    A, ADDL       ;SEND LOW ADD
        X     A, SIOR       ;
        SBIT  BUSY, PSW     ;
LP5:    IFBIT BUSY, PSW     ;
        JP    LP5           ;
        LD    SIOR, #000    ;
        SBIT  BUSY, PSW     ;RECEIVE BYTE
        RBIT  BUSY, PSW     ;
        SBIT  BUSY, PSW     ;
LP6:    IFBIT BUSY, PSW     ;
        JP    LP6           ;
        X     A, SIOR       ;
        X     A, BYT        ;
        LD    PORTD, #00    ;
        JSR   COUNT         ;
        LD    A, HLD        ;
        IFEQ  A, STOHLD     ;
        JP    SKIP1         ;
        RET                 ;
SKIP1:  LD    A, ADDL       ;
        IFEQ  A, STOLO      ;
        JP    SKIP2         ;
        RET                 ;
SKIP2:  LD    A, ADDH       ;
        IFEQ  A, STOHI      ;
        JP    ZD            ;
        RET                 ;
;
PUT:    LD    A, HLD        ;SET CS HIGH
        X     A, PORTD      ;
        LD    A, ADDH       ;SEND OPPCOCE AND
        OR    A, #028       ;HI ADDRESS
        X     A, SIOR       ;
        SBIT  BUSY, PSW     ;
LP7:    IFBIT BUSY, PSW     ;
        JP    LP7           ;
```

TL/D/12033–5

```
        LD    A, ADDL       ;SEND LOW ADD
        X     A, SIOR       ;
        SBIT  BUSY, PSW     ;
LP8:    IFBIT BUSY, PSW     ;
        JP    LP8           ;
        LD    A, BYT        ;SEND BYTE
        X     A, SIOR       ;
        SBIT  BUSY, PSW     ;
LP9:    IFBIT BUSY, PSW     ;
        JP    LP9           ;
LP10:   IFBIT SI, PORTGP    ;
        JP    LP10          ;
LP11:   IFBIT SI, PORTGP    ;
        JP    LP12          ;
        JP    LP11          ;
LP12:   LD    PORTD, #00    ;
        LD    A, HLD        ;
        X     A, PORTD      ;
        LD    SIOR, #0FF    ;
        SBIT  BUSY, PSW     ;
        RBIT  BUSY, PSW     ;
        LD    PORTD, #00    ;
        JSR   COUNT         ;
        RET                 ;
;
EWEN:   LD    PORTD, #0F0   ; Enable EE
        LD    SIOR, #026    ;
        SBIT  BUSY, PSW     ;
LP13:   IFBIT BUSY, PSW     ;
        JP    LP13          ;
        LD    SIOR, #00     ;
        SBIT  BUSY, PSW     ;
LP14:   IFBIT BUSY, PSW     ;
        JP    LP14          ;
        LD    PORTD, #00    ;
        RET                 ;
;
COUNT:  LD    A, ADDL       ; Address Counter
        IFEQ  A, #0FF       ;
        JP    ZA            ;
        INC   A             ;
        X     A, ADDL       ;
        RET                 ;
ZA:     LD    A, ADDH       ;
        IFEQ  A, #07        ;
        JP    ZC            ;
        INC   A             ;
        X     A, ADDH       ;
        LD    ADDL, #00     ;
        RET                 ;
ZC:     LD    A, HLD        ;
        IFEQ  A, #080       ;
        JP    ZE            ;
        LD    A, HLD        ;
        ADD   A, HLD        ;
        X     A, HLD        ;
        LD    ADDL, #00     ;
        LD    ADDH, #00     ;
        RET                 ;
ZE:     JSR   SAVE          ;
```

TL/D/12033–6

```
ZD:     LD   PORTD, #02     ;
        JSR  DELAY          ;
        LD   PORTD, #00     ;
        JSR  DELAY          ;
        JP   ZD             ;
;
SAVE:   LD   A, ADDL        ; Save Pointer
        X    A, STOLO       ;
        LD   ADDL, #0FD     ;
        LD   A, ADDH        ;
        X    A, STOHI       ;
        LD   ADDH, #07      ;
        LD   A, HLD         ;
        X    A, STOHLD      ;
        LD   HLD, #084      ;
        LD   A, STOLO       ;
        X    A, BYT         ;
        JSR  PUT            ;
        LD   A, STOHI       ;
        X    A, BYT         ;
        JSR  PUT            ;
        LD   A, STOHLD      ;
        X    A, BYT         ;
        JSR  PUT            ;
        IFBIT 3, [B]        ;
        JSR   GOTIT         ;
        RET

GOTIT:  LD   PORTD, #02     ; LED Flashing sequence
        JSR  DELAY          ;
        LD   PORTD, #00     ;
        JSR  DELAY          ;
        LD   PORTD, #02     ;
        JSR  DELAY          ;
        LD   PORTD, #00     ;
        JSR  DELAY          ;
        JSR  DELAY          ;
        JSR  DELAY          ;
        JP   GOTIT          ;
;
LAST:   LD   ADDL, #0FD     ; Get pointer
        LD   ADDH, #07      ;
        LD   HLD, #084      ;
        JSR  GET            ;
        LD   A, BYT         ;
        X    A, STOLO       ;
        JSR  GET            ;
        LD   A, BYT         ;
        X    A, STOHI       ;
        JSR  GET            ;
        LD   A, BYT         ;
        X    A, STOHLD      ;
        LD   ADDL, #00      ;
        LD   ADDH, #00      ;
        LD   HLD, #010      ;
        RET                 ;
;
FLSH2:  LD    PORTD, #001   ;FLASH LED 2 TIMES
        JSR   DELAY         ;
        LD    PORTD, #00    ;
```

TL/D/12033–7

```
        JSR   DELAY          ;
        LD    PORTD, #001    ;
        JSR   DELAY          ;
        LD    PORTD, #00     ;
        RET                  ;
;
FLSH1:  LD    PORTD, #001    ;FLASH LED ONCE
        JSR   DELAY          ;
        LD    PORTD, #00     ;
        JSR   DELAY          ;
        RET                  ;
;
DELAY:  LD   DLYH, #040      ;
LP15:   LD   DLYL, #0FF      ;
LP16:   DRSZ DLYL            ;
        JP   LP16            ;
        DRSZ DLYH            ;
        JP   LP15            ;
        RET                  ;
        .END  START          ;
```

TL/D/12033–8

How to Use the NM93C86A Serial EEPROM as a PC/Laptop Detachable Printer File Memory Card (DPFMC)

AN-936

LIFE SUPPORT POLICY

NATIONAL'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS WRITTEN APPROVAL OF THE PRESIDENT OF NATIONAL SEMICONDUCTOR CORPORATION. As used herein:

1. Life support devices or systems are devices or systems which, (a) are intended for surgical implant into the body, or (b) support or sustain life, and whose failure to perform, when properly used in accordance with instructions for use provided in the labeling, can be reasonably expected to result in a significant injury to the user.
2. A critical component is any component of a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness.

National Semiconductor Corporation
2900 Semiconductor Drive
P.O. Box 58090
Santa Clara, CA 95052-8090
Tel: 1(800) 272-9959
TWX: (910) 339-9240

National Semiconductor GmbH
Livry-Gargan-Str. 10
D-82256 Fürstenfeldbruck
Germany
Tel: (81-41) 35-0
Telex: 527649
Fax: (81-41) 35-1

National Semiconductor Japan Ltd.
Sumitomo Chemical
Engineering Center
Bldg. 7F
1-7-1, Nakase, Mihama-Ku
Chiba-City,
Ciba Prefecture 261
Tel: (043) 299-2300
Fax: (043) 299-2500

National Semiconductor Hong Kong Ltd.
13th Floor, Straight Block,
Ocean Centre, 5 Canton Rd.
Tsimshatsui, Kowloon
Hong Kong
Tel: (852) 2737-1600
Fax: (852) 2736-9960

National Semiconductores Do Brazil Ltda.
Rue Deputado Lacorda Franco
120-3A
Sao Paulo-SP
Brazil 05418-000
Tel: (55-11) 212-5066
Telex: 391-1131931 NSBR BR
Fax: (55-11) 212-1181

National Semiconductor (Australia) Pty, Ltd.
Building 16
Business Park Drive
Monash Business Park
Nottinghill, Melbourne
Victoria 3168 Australia
Tel: (3) 558-9999
Fax: (3) 558-9998