MPC8260 Reset and Configuration Word

by *Paul Genua, P.E.*
NCSG Field Applications
Freescale Semiconductor, Inc.

This application note describes how to design for the 82xx Hardware Configuration Word fetch through programmable logic, as opposed to the EEPROM originally described in the *MPC8260 PowerQUICC II™ User's Manual*.

Contents

1 Introduction

With the advent of the MPC8260, Freescale Semiconductor has introduced a new method for providing the processor with initial hardware configuration data. Upon Power-On-Reset (POR) the 82xx is designed to fetch a 32 bit word from the 60x bus in four 8-bit accesses. Though this was originally intended to be provided via non-volatile memory, such as described in the 82xx databook, it is possible to provide the hardware configuration word through a PLD or FPGA located on the 60x bus. This allows the user to change the hardware configuration word without using the same NV-RAM for both boot code and hardware configuration word storage. This document is intended to give the designer examples of different possibilities for hardware configuration of the 82xx through programmable logic.

2 MPC8260 Hardware Configuration Sequence

As described in the processor reference manual, assertion of Power-On-Reset (POR) starts the power-on-reset flow and

initializes the 82xx. Upon deassertion of POR, the 82xx reads the state of ~RSTCONF. If ~RSTCONF is detected as a logic-high state upon deassertion of POR, then the 82xx becomes a configuration slave and waits for toggling of ~RSTCONF to latch in a hardware reset configuration word from its data bus. If ~RSTCONF is logic-low upon deassertion of POR, then the 82xx determines that it is a configuration master and then initializes accesses to fetch its hardware configuration word.

The Configuration word for the now master 82xx is accessed as four, byte-wide transfers to a memory located at CS0 on the 60x Bus.The hardware word is read, most significant byte first, from the most significant byte lane on the 82xx.

Table 1. MPC8260 Reset Word Accesses

Byte 0 Address	Byte 1 Address	Byte 2 Address	Byte 3 Address
0x00	0x08	0x10	0x18

The master 82xx attempts to configure up to 7 slave 82xx devices. It continues reading 4 more bytes in order to form the configuration word of a first slave device. The Master 82xx then reassembles the word into one 32-bit word and drives the word onto the Slave's D[0-31] while toggling its RSTCONF as a strobe for latching in the hardware word. This will continue until the Master reads and configures slaves 1 through 7. See Figure 1 and Figure 2.

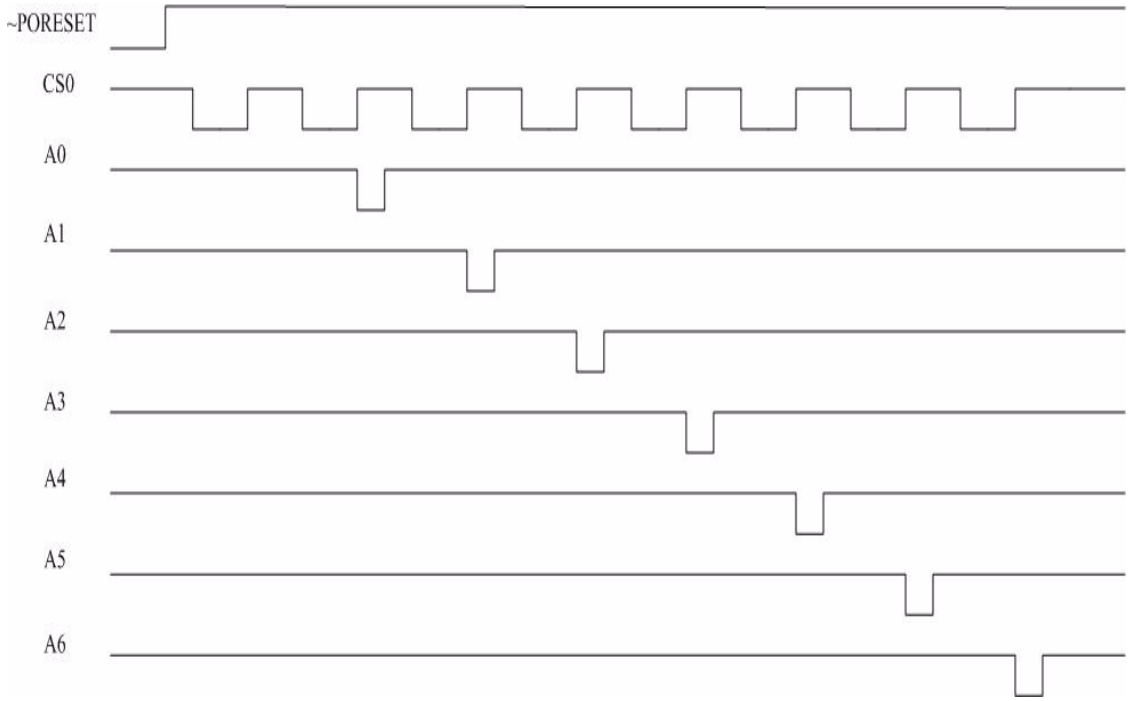

Figure 1. MPC82xx Reset Configuration Sequence

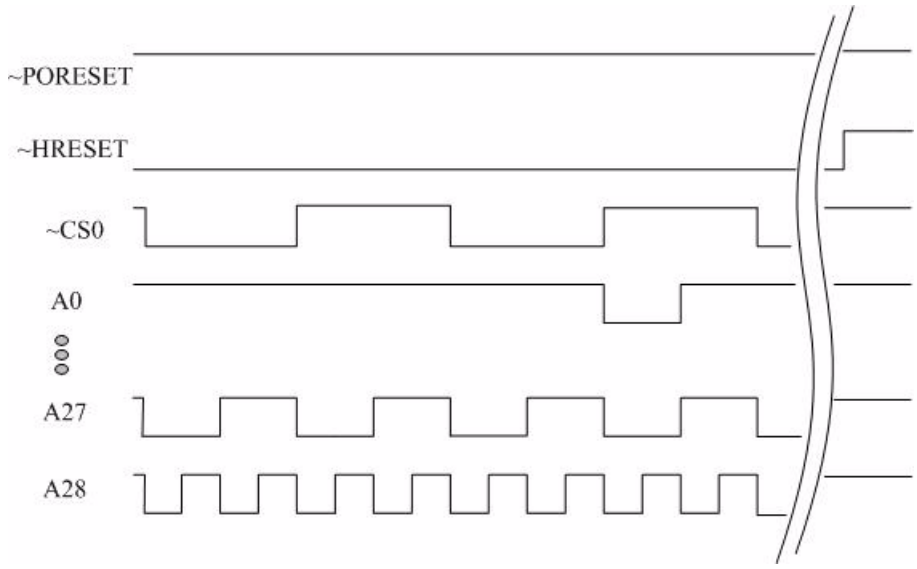

Figure 2. Master Configuration of 1st Slave Device

The Hardware Configuration Word sequence will be initiated at every ~Hreset assertion.

3 MPC8620 Slave Configuration

A system can be designed with a PLD emulating the intended master MPC8260, such that the MPC8260 in question can be configured as a slave. A typical schematic is shown in Figure 3.

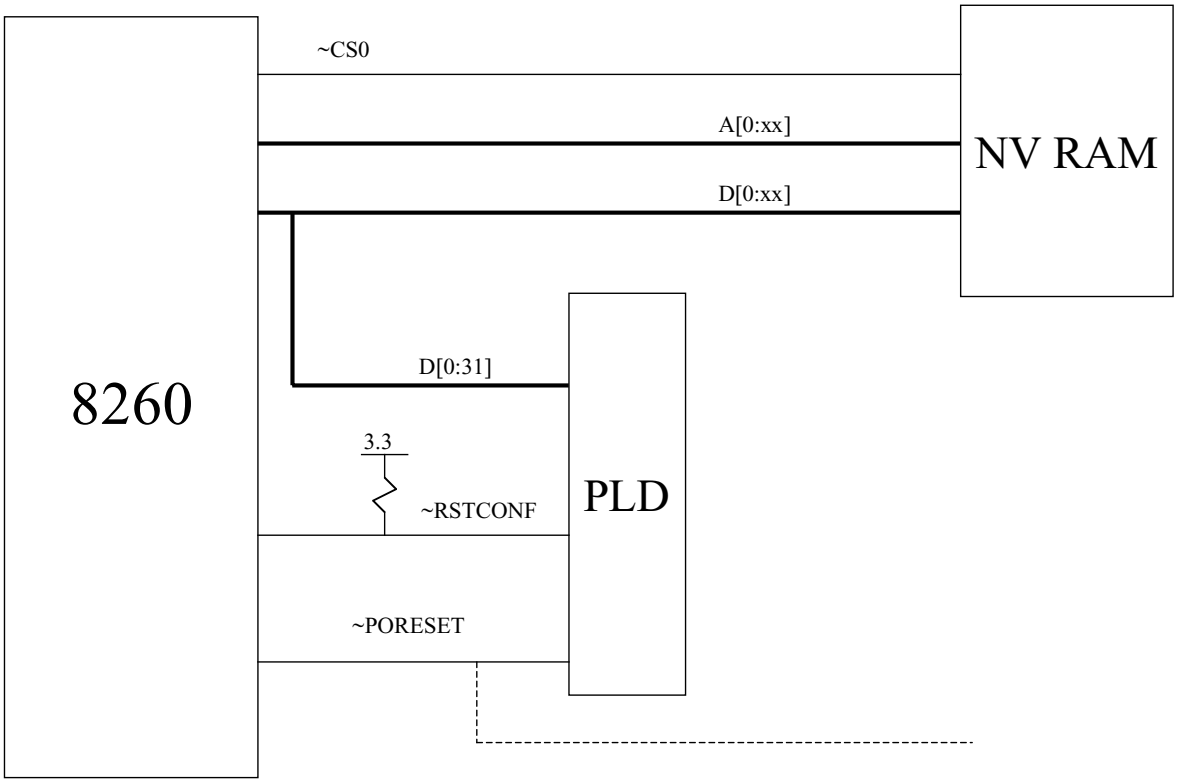

Figure 3. MPC8260 Slave Configuration

The PLD, in the slave configuration mode, will need to drive the whole configuration word onto the 82xx's data bus while toggling the ~RSTCONF line. **Note that the 82xx needs a minimum of 5ns from deassertion of ~PORESET to driving ~RSTCONF to ensure the part does not come up in master mode.**

The slave 82xx then will latch the reset configuration word from the 60x bus on the rising edge of the system clock. **If the ~RSTCONF pulse spans multiple clock cycles, then the value of the data bus on the last clock's rising edge will be the one used for the reset configuration word.** Thus one must ensure the ~RSTCONF pulse is greater than a clock period plus setup/hold time.

The configuration process must be completed before the MODCK pins are sampled by the processor (1024 clock cycles after ~PORESET is deasserted). If ~RSTCONF is not toggled by 1024 clocks after deassertion of ~PORESET, then the 8260 will boot with the default configuration as described in the reference manual.

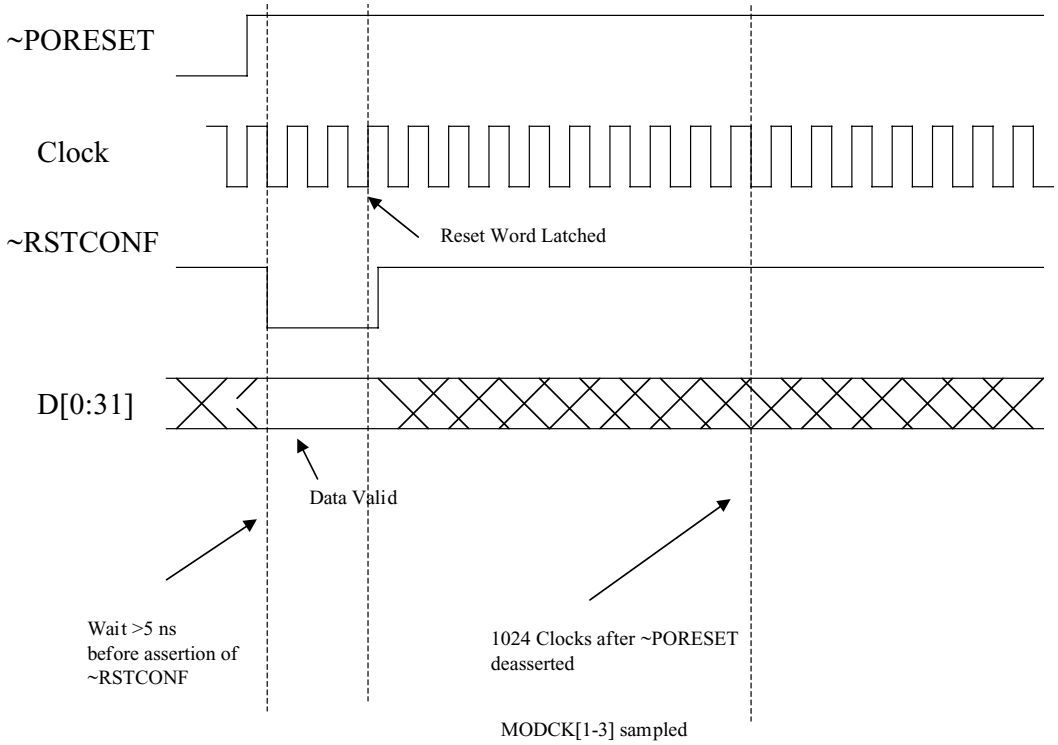

Figure 4. Slave Configuration Timing

4 MPC8260 Master Configuration

It may be desirable to design a system in which a PLD is used to emulate the intended EPROM for 82xx Master Configuration Setup. In such a design, the PLD would sit on the bus between the 8260 and nonvolatile memory, in effect intercepting Chip Select 0 (CS0) during the configuration phase and passing CS0 through to NV-RAM during normal operation.

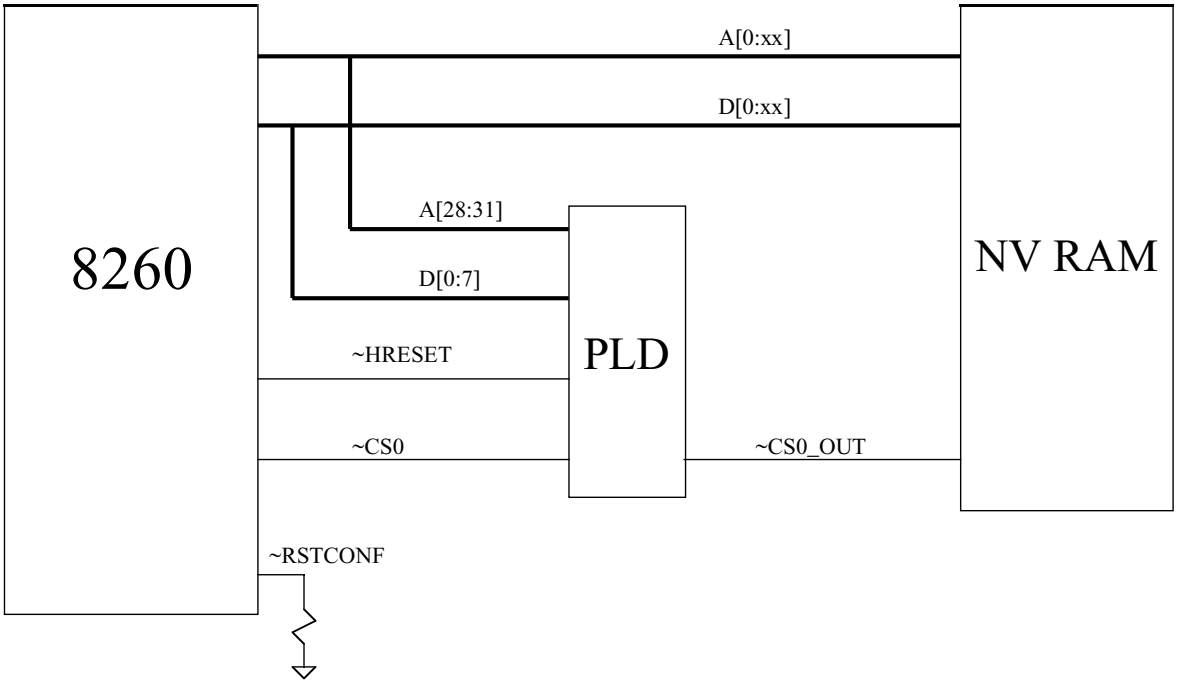

Figure 5. MPC8260 Master Configuration

The 82xx, in Master configuration mode, will fetch the hardware configuration word from ~CS0 in four, byte-wide transfers. The PLD will act as a 4-byte device, to be accessed on ~CS0. If the master is not used to configure slave devices, only addresses A[27:28] are necessary at the PLD. Since the four configuration bytes are located at addresses 0x00, 0x08, 0x10, and 0x18, this would save pins on the external device. In the PLD logic we are able to use ~HRESET AND'ed with CS0 to select the PLD's register for the hardware configuration word since ~HRESET is always asserted during the configuration process.

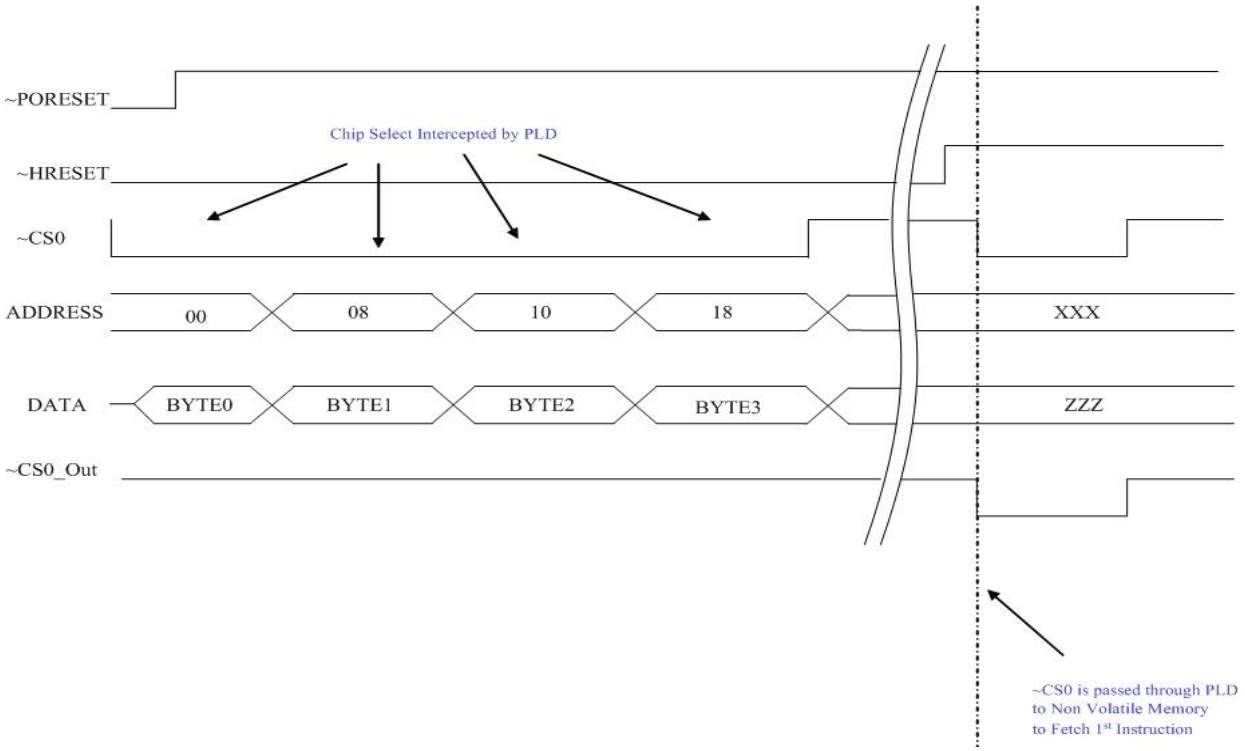

Figure 6. Master Configuration Timing

5 Sample Master Configuration VHDL Code

File Name: master.vhd

Rev: 0.0

Author: Paul Genua

This code is meant to provide a hardware configuration word of 0x0C820205 to an MPC8260 configuration master during the boot-up sequence.

CS0 and HRESETN are used to determine that the configuration word is to be provided. In the absence of HRESETN low, CS0_IN is passed through the PLD to CS0_OUT which is connected to NVRAM.

```
library ieee;

use ieee.std_logic_1164.all;


package pkgmaster is

    component master

    port (

                hresetn          : in STD_LOGIC;

                        -- hresetn: 8260 hard reset

                cs0_in           : in STD_LOGIC;
```

```
                -- CS0 output from 8260
        cs0_out         : out STD_LOGIC;
                -- CS0 output from PLD to NVRAM

        data860_out_p: out STD_LOGIC_VECTOR (7 downto 0);
                -- 8260 DataBus
        addr860         : in  STD_LOGIC_VECTOR (4 downto 0)
                -- 8260 Address Bus
        );
    end component;
end pkgmaster;


library ieee;
use ieee.std_logic_1164.all;

entity master is
        port (
                hresetn         : in STD_LOGIC;
                        -- hresetn: 8260 hard reset
                cs0_in          : in STD_LOGIC;
                        -- CS0 output from 8260
                cs0_out         : out STD_LOGIC;
                        -- CS0 output from PLD to NVRAM
                data860_out_p: out STD_LOGIC_VECTOR (7 downto 0);
                        -- 8260 DataBus
                addr860         : in  STD_LOGIC_VECTOR (4 downto 0)
                        -- 8260 Address Bus
        );
end master;

architecture master_arch of master is
```

```
        signal data860_z: STD_LOGIC_VECTOR (7 downto 0);
        signal data860_in: STD_LOGIC_VECTOR (7 downto 0);

        signal data_cs0 : std_logic_vector (7 downto 0);

        signal csn_d0   : std_logic;

        signal wr_cs0   : boolean;

begin

-- CS0 Process
-- When Hreset = 1 CS0_in gets passed through to CS0_out for FLASH
-- When resetting, acts as ROM & outputs 8260 HRESET config
-- HRESET Word is four byte accesses at 0x00, 0x08, 0x10, 0x18
process begin
                if cs0_in = '0' then
                        if hresetn= '1' then
                                cs0_out <= '0';
                                wr_cs0 <= false ;
                        else
                                        cs0_out <='1';
                                        wr_cs0 <= true;
                                        case addr860 is
                                                when "00000" =>
                                                        data_cs0 <= "00001100";
                                                when "01000" =>
                                                        data_cs0 <= "10000010";
                                                when "10000" =>
                                                        data_cs0 <= "00000010";
                                                when "11000" =>
```

```
                                                    data_cs0 <= "00000101";
                                                when others=>
                                                    data_cs0 <= "00000000";
                                            end case;
                        end if;
                else if cs0_in = '1' then
                        cs0_out <='1';
                        wr_cs0 <= false ;
                end if;
                end if;
end process;


data860_out_p <= data_cs0 when wr_cs0 else (others => 'Z');


end master_arch;
```

6 Revision History

Table 2. Revision History

Revision	Release Date	Changes
1	11/2004	Updated Figure 2, Figure 5, and Figure 6. Added Figure 1. Updated text in Section 2, “MPC8260 Hardware Configuration Sequence,” in order to clarify the Master Configuration cycle.
0	10/2002	Initial release

THIS PAGE INTENTIONALLY LEFT BLANK

How to Reach Us:

Home Page:
www.freescale.com

email:
support@freescale.com

USA/Europe or Locations Not Listed:
Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
(800) 521-6274
480-768-2130
support@freescale.com

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
support@freescale.com

Japan:
Freescale Semiconductor Japan Ltd.
Technical Information Center
3-20-1, Minami-Azabu, Minato-ku
Tokyo 106-0047 Japan
0120 191014
+81 3 3440 3569
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate,
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor
Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
(800) 441-2447
303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@
hibbertgroup.com

AN2349
Rev. 1
11/2004

freescale™
semiconductor